ProAct: Agentic Lookahead in Interactive Environments

Yangbin Yu*, Mingyu Yang*, Junyou Li, Yiming Gao, Feiyu Liu, Yijun Yang, Zichuan Lin, Jiafei Lyu, Zhicong Lu, Deheng Ye, and Jie Jiang†

Tencent Hunyuan

Abstract. LLM agents often fail in long-horizon tasks due to compounding simulation errors. We propose ProAct, a two-stage framework for internalizing foresight. First, Grounded LookAhead Distillation (GLA D) performs supervised fine-tuning (SFT) on search-derived trajectories, compressing complex search trees into concise causal reasoning chains. This allows agents to learn lookahead logic without inference-time computational overhead. Second, we introduce Monte-Carlo Critic (MC-Critic), a plug-and-play auxiliary value estimator for policy-gradient algorithms (PPO/GRPO). By using lightweight rollouts to provide low-variance value signals, MC-Critic enables stable optimization without expensive model-based approximations. Experiments on stochastic (2048) and deterministic (Sokoban) tasks show ProAct significantly boosts planning accuracy. Notably, a 4B model trained with ProAct outperforms open-source baselines and rivals state-of-the-art closed-source models, demonstrating strong generalization to unseen environments. The codes and models are available at `https://github.com/GreatX3/ProAct`.

Keywords: Large Language Model Agents · Reasoning · Agentic Reinforcement Learning

1 Introduction

Recent advancements in Large Language Models (LLMs) have enabled autonomous agents to tackle complex, interactive tasks [12, 20, 35, 39–41, 55, 60]. Because these environments require sequential decision-making, optimal performance demands human-like System 2 processing [6, 11, 16, 31]—mentally simulating and comparing future trajectories before acting.

However, replicating this lookahead capability remains a fundamental challenge. While methods like Chain-of-Thought [37, 47] and ReAct [48] improve reasoning, they struggle with compounding simulation errors in long-horizon tasks [17]. Minor inaccuracies in predicting environmental dynamics rapidly accumulate [17, 20], causing plans to diverge from reality. Consequently, extending ungrounded reasoning exacerbates hallucinations and context drift, while single-step reasoning fails to account for long-term consequences.

* Equal contribution.
† Corresponding author. Email: zeus@tencent.com

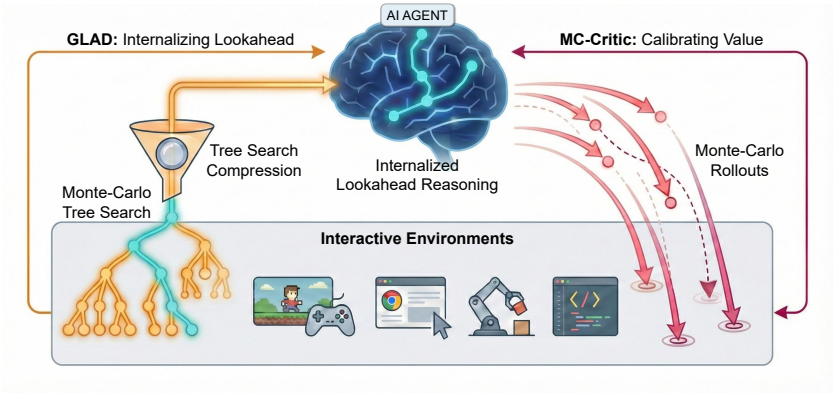

Fig. 1: Overview of ProAct. A two-stage paradigm to internalize accurate lookahead reasoning for AI agents. GLAD distills complex MCTS search trees into concise, causal reasoning chains via SFT. MC-Critic leverages lightweight environment rollouts to provide low-variance value estimates, stabilizing online RL training.

To bridge this gap, we propose ProAct, a two-stage framework that internalizes accurate, multi-turn lookahead capabilities into LLM agents via grounded supervision and environment-augmented reinforcement learning (Fig. 1). First, during Supervised Fine-Tuning (SFT), we introduce Grounded LookAhead Distillation (GLAD). We perform environment-driven Monte-Carlo Tree Search (MCTS) to explore trajectories, compressing these expansive search trees into concise, high-quality reasoning chains. Training explicitly on these grounded chains teaches the model to predict outcomes and identify risks using true environmental feedback, mitigating hallucinated physics. Second, we introduce a Reinforcement Learning (RL) phase augmented by a Monte-Carlo Critic (MC-Critic). To resolve value estimation struggles in standard multi-turn RL [13, 27, 28], MC-Critic leverages lightweight environmental rollouts to provide stable, low-variance value estimates, efficiently aligning the agent's internalized lookahead with reality.

We evaluate ProAct on rigorous long-horizon games (e.g., 2048, Sokoban). Experiments demonstrate that a 4B parameter ProAct model significantly enhances multi-turn decision-making, outperforming all open-source baselines and achieving performance comparable to state-of-the-art closed-source models, alongside strong zero-shot generalization to unseen environments.

Our contributions are summarized as follows:

1. We propose **GLAD**, a method to internalize environmental dynamics into LLM agents by compressing search-based trajectories into explicit reasoning chains, mitigating simulation hallucinations.
2. We introduce **MC-Critic**, a plug-and-play auxiliary value estimation method that leverages environment interaction to stabilize and accelerate multi-turn agentic RL training.

3. We demonstrate that a 4B parameter model trained with **ProAct** outperforms all open-source baselines and achieves performance comparable to state-of-the-art closed-source models. Furthermore, the model demonstrates strong generalization to unseen environments.

2 Related Work

Recent research in LLM agents has shifted from static, single-turn problem solving to dynamic, multi-turn interactions in complex environments. This paradigm shift necessitates advancements across three critical dimensions: (1) Multi-Turn Agentic RL, which addresses the stability and exploration challenges inherent in trajectory-level optimization; (2) Reasoning and Search Distillation, where expensive inference-time planning (System 2) is compressed into efficient policy intuition (System 1); and (3) Value Estimation for Agentic RL, which tackles the difficulty of attributing sparse rewards to specific reasoning steps in long-horizon tasks. Our work synthesizes these directions, proposing a unified framework that calibrates agent reasoning via environment-based search and stabilizes policy updates through Monte-Carlo value estimation.

Multi-Turn Agentic Reinforcement Learning Training LLMs as agents requires moving beyond single-turn alignment to trajectory-level optimization. Recent frameworks like AgentGym-RL [40] and RAGEN [36] have established the foundations for this transition. AgentGym-RL [40] introduces ScalingInter-RL, a curriculum strategy that progressively expands interaction horizons to prevent model collapse during exploration. Similarly, RAGEN [36] identifies the "Echo Trap" phenomenon in multi-turn RL where agents overfit to shallow reasoning patterns and proposes StarPO to stabilize trajectory-level updates. Other systems like SkyRL [5] and DART [15] focus on the asynchronous execution efficiency of these long-horizon rollouts, while many other works [8, 18, 38, 44, 53], etc., further improve the stability and efficiency of multi-turn agentic RL. While these frameworks [33] provide the infrastructure for agent training, our work focuses on the quality of the internal reasoning process of agent. Unlike methods that rely on implicit rewards or pure trial-and-error, we introduce an environment-calibrated reasoning paradigm that actively mitigates the cumulative errors inherent in long-horizon internal simulations.

Reasoning Distillation from System 2 to System 1 Enhancing agents with "System 2" capabilities is a central theme in recent research. Methods like Tree of Thoughts [47] and RAP [7] integrate explicit search algorithms (e.g., BFS, MCTS) during inference to explore reasoning paths. However, the high computational cost of inference-time search limits their scalability [10, 14, 51]. Consequently, research has pivoted towards distillation: transferring the capabilities of expensive planners into efficient "System 1" policies. Distilling Step-by-Step [9] and STaR [52] demonstrate that LLMs can bootstrap their own reasoning capabilities by fine-tuning on self-generated rationales. Furthermore, VAGEN [34]

explicitly enforces the generation of internal "world models" (e.g., state estimation and dynamics simulation) to ground agent reasoning, and WALL-E [60] aligns the agent's world model with environment dynamics via neurosymbolic learning to improve world model-based LLM agents. Our approach advances this direction by introducing Reasoning Compression. Instead of simply cloning verbose search traces or explicit world model states, we use MCTS to calibrate the agent's thought process against ground-truth environmental dynamics, and then compress this search tree into a concise, natural language estimation of future trends. This effectively distills the foresight of MCTS (System 2) into a token-efficient policy (System 1) that maintains diversity and accuracy without the overhead of explicit search tags.

Value Estimation in Agentic Reinforcement Learning Accurate value estimation is critical for guiding exploration in sparse-reward environments, yet it remains a bottleneck for RL agents. Traditional neural critics [23, 27, 54, 58, 59] often suffer from high bias and slow convergence in high-dimensional language spaces. To address this, ArCHer [62] proposes a hierarchical architecture that estimates value at the utterance level to improve credit assignment over long horizons. SWEET-RL [61] introduces an asymmetric critic with access to privileged training-time information to provide dense step-level rewards. Turn-Level Reward Design [13,19] further explores granular feedback mechanisms to extend algorithms like GRPO [28] to multi-turn settings. Diverging from these parametric approaches, our method introduces a Monte-Carlo Critic. Instead of training a separate critic network, we leverage low-cost Monte-Carlo rollouts to estimate state values dynamically. This provides a low-variance, environment-grounded signal that significantly enhances the stability of policy gradient methods (e.g., PPO [27], GRPO [28]).

3 The ProAct Framework

3.1 Overview

Formulation We consider the problem of an LLM agent interacting with environments, formalized as a Markov Decision Process (MDP) tuple $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma \rangle$. At each time step t, the agent observes a state $s_t \in \mathcal{S}$, selects an action $a_t \in \mathcal{A}$, receives a step-level reward $r_t = \mathcal{R}(s_t, a_t)$, and the environment transitions to a new state $s_{t+1} \sim \mathcal{P}(\cdot|s_t, a_t)$. The objective is to learn a policy π_θ parameterized by θ that maximizes the expected cumulative discounted return $J(\theta) = \mathbb{E}_{\tau \sim \pi_\theta}[\sum_{t=0}^{\infty} \gamma^t r_t]$.

Unlike traditional RL agents [21, 22, 29, 30, 42] that directly map states to actions, LLM agents possess the capability to perform explicit reasoning before acting. We formulate the policy of LLM agents as a joint distribution over a reasoning chain z_t and an action a_t:

$$\pi_\theta(z_t, a_t|s_t) = \pi_\theta(z_t|s_t) \cdot \pi_\theta(a_t|s_t, z_t), \tag{1}$$

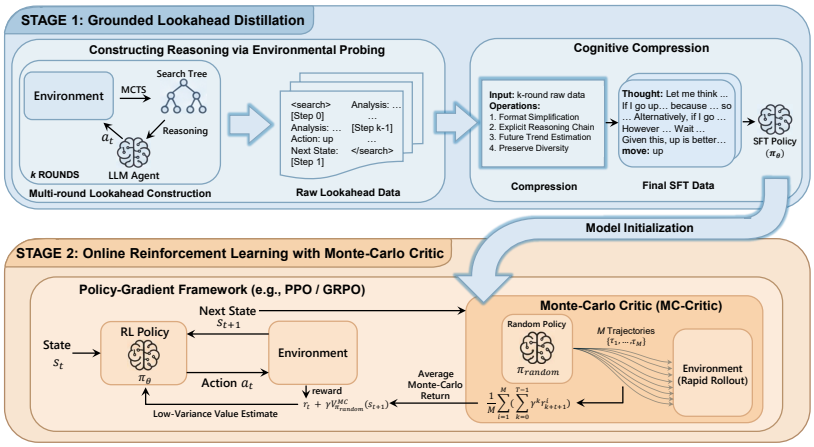

Fig. 2: ProAct internalizes lookahead reasoning in two stages. Stage 1 (Grounded LookAhead Distillation) distills complex MCTS search trees into compressed, explicit reasoning chains. Stage 2 (Online RL with MC-Critic) refines reasoning accuracy via policy gradients (e.g., PPO/GRPO). This plug-and-play critic provides low-variance value estimates by aggregating discounted returns from M parallel trajectories.

where z_t represents a sequence of intermediate reasoning tokens (e.g., state analysis, future prediction, sub-goal decomposition) generated by the LLM. We represent all output tokens as c_t, i.e., $c_t = \langle z_t, a_t \rangle$. This formulation decomposes the decision-making process into two stages: *deliberation* ($\pi_\theta(z|s)$) and *execution* ($\pi_\theta(a|s,z)$).

Our Approach In long-horizon interactive environments, optimal decision-making often requires looking ahead multiple steps into the future. An LLM agent attempts to emulate this by generating a reasoning chain z_t that implicitly models the environment's transition dynamics. However, a critical discrepancy arises between the agent's internal world model and the actual environment dynamics. If the agent generates a reasoning chain based on a hallucinated transition $P_{model}(s'|s,a) \neq \mathcal{P}(s'|s,a)$, the resulting plan becomes invalid. As the lookahead depth increases, these errors compound exponentially, a phenomenon we term *simulation drift*. To address this, we propose ProAct, a framework that grounds the internal reasoning of agent in external reality, as shown in Fig. 2. We aim to optimize π_θ such that the generated reasoning z_t accurately reflects the true future value of actions without requiring expensive searching at inference time. This is achieved through two phases:

1. **Grounded LookAhead Distillation:** We supervise the deliberation policy $\pi_\theta(z|s)$ using compressed trajectories derived from a search algorithm (e.g., MCTS) interacting with the environment $\mathcal{P}$.

2. **Monte-Carlo Critic:** We further refine the agent using online RL, where the value estimation is calibrated by a Monte-Carlo Critic that offers low-variance advantage estimation, ensuring stable RL training.

3.2 Grounded LookAhead Distillation

The intuition of GLAD is that long-horizon decision-making of interactive environments relies on looking ahead, simulating potential futures before committing to an action. However, LLM agents suffer from simulation drift, where their internal world models diverge from reality over long horizons. Instead of relying on the internal hallucinations of agents, we propose to externalize the lookahead process during reasoning data generation. We construct reasoning trajectories by allowing the model to read real future outcomes provided by the environment, and then compress this explicit search into an implicit intuition for future trends.

Constructing Reasoning via Environmental Probing To generate high-quality supervision data, we design an iterative interaction loop where the environment acts as an oracle to ground the reasoning process of the model. This process consists of N rounds of deliberation for a single decision:

Algorithm 1 ProAct: Grounded Lookahead Distillation

Input: Environment $\mathcal{E}$, Agent Policy π_θ, Random Sampler
Output: Dataset Buffer $\mathcal{D} \leftarrow \emptyset$
for episode $e = 1$ to B **do**
  Initialize state s_0, History $H \leftarrow \emptyset$
  $t \leftarrow 0$
  while not *Done* **do**
    $\mathcal{T}_{\text{real}} \leftarrow \text{ProbeEnvironment}(s_t, \text{depth} = T, \text{samples} = N)$
    $raw\_analysis, a_t \leftarrow \pi_\theta(\text{Prompt}(s_t, \mathcal{T}_{\text{real}}))$
    if $a_t ==$ `<BACKTRACK>` **then**
      Revert s_t to previous state s_{t-1} (if applicable)
      Continue to next iteration
    else
      Execute action: $s_{t+1}, r_t \leftarrow \mathcal{E}.\text{step}(a_t)$
      Store interaction: $H.\text{append}((s_t, \mathcal{T}_{\text{real}}, raw\_analysis, a_t))$
      $t \leftarrow t + 1$
    end if
  end while
  for each step $(s_t, \mathcal{T}_{\text{real}}, raw\_analysis, a_t)$ in H **do**
    $z_{\text{compressed}} \leftarrow \text{Compress}(s_t, \mathcal{T}_{\text{real}}, a_t)$
    Add to dataset: $\mathcal{D} \leftarrow \mathcal{D} \cup \{(s_t, z_{\text{compressed}}, a_t)\}$
  end for
end for

Step 1: Environment-Augmented Lookahead. At each reasoning step t, rather than letting the model guess the future, we explicitly probe the real environment. We execute a MCTS starting from the current state s_t. We sample N trajectories $\{\tau_1, ..., \tau_N\}$, where each trajectory $\tau_i = (s_t, a_t^i, r_t^i, s_{t+1}^i, ...)$ extends for T steps. Crucially, we record both optimal and suboptimal/dead-end paths. These trajectories serve as a "ground-truth future map."

Step 2: Trajectory-Aware Decision Making. We feed the current state s_t and the sampled raw trajectories directly into the context of LLM agent. The model is prompted to analyze and simulate these futures (e.g., "Trajectory A leads to a merge, while Trajectory B leads to a gridlock"). Based on this unbiased environmental feedback, the model outputs:

1. **Analysis:** A comparison and simulation of the potential futures.
2. **Decision:** The next action to take, or a special token `<BACKTRACK>` if the analysis reveals that the current branch is suboptimal compared to previous alternatives.

This "Probing-Decision-Reflection" loop allows the model to correct its course if it identifies a bad state, effectively simulating a deep, self-correcting thought process grounded in reality.

Cognitive Compression The raw interaction contexts from Phase 1 contain verbose search traces, structural tags, and backtracking steps. Directly fine-tuning on this data is inefficient and prone to overfitting. We introduce a compression step to distill the explicit search into a concise reasoning chain that estimates future trends.

We employ a teacher model (or the model itself) to synthesize the raw contexts into a final reasoning path z, adhering to four strict principles:

1. **Format Simplification:** We remove all structural artifacts. The reasoning is rewritten in natural language (e.g., "Let's analyze the board...", "If I move up, the tiles will merge...") to align with the pre-trained distribution of LLM.
2. **Explicit Reasoning Chains:** Each step must follow a strict causal logic: *Observation* $\rightarrow$ *Analysis* $\rightarrow$ *Conclusion*. The analysis must explicitly link current actions to future states based on the rules of environment observed in Phase 1.
3. **Future Trend Estimation:** The compressed reasoning must not only explain the chosen action but also explain why other actions were rejected. For example, "Moving left is safe now but blocks a critical merge in the future." This forces the model to internalize the environmental dynamics and learn counterfactual reasoning.
4. **Preserve Diversity:** The reasoning z must retain the "trade-off" analysis found in the search. Instead of dogmatically stating the answer, it should reflect the deliberation process (e.g., "Option A is good for score, but Option B provides better safety. Considering the long term, I choose B."), preserving the diversity of thought.

Finally, we perform SFT on the dataset $\mathcal{D} = \{(s, z_{compressed}, a)\}$, minimizing the standard negative log-likelihood loss. This process effectively teaches the model to hallucinate correct future trends without needing to perform expensive search at inference time. We show the pseudocode of GLAD in Algorithm 1.

3.3 Online RL with Monte-Carlo Critic

RL has recently become a prominent method for empowering LLM agents [6,11]. However, despite its efficacy, applying RL to LLMs presents significant hurdles, particularly regarding value function (i.e., critic network) estimation in long-horizon scenarios. Formally, the state-value function $V_\pi(s_t)$ represents the expected cumulative reward an agent obtains starting from state s_t and following policy π. Similarly, the action-value function $Q_\pi(s_t, a_t)$ denotes the expected cumulative reward starting from state s_t, taking action a_t, and thereafter following policy π. These two value functions are defined as follows:

$$V_\pi(s_t) = \mathbb{E}_{\tau\sim\pi}\left[\sum_{k=t}^{\infty}\gamma^{k-t}r_k\right], \tag{2}$$

$$Q_\pi(s_t, a_t) = \mathbb{E}_{s_{t+1}\sim\mathcal{P}(\cdot|s_t,a_t)}\left[r_t + \gamma V_\pi(s_{t+1})\right]. \tag{3}$$

In traditional deep RL [45,46,56,57], policy networks typically consist of simple Multi-Layer Perceptrons (MLPs) with millions of parameters. This lightweight architecture permits rapid environment interaction, allowing agents to quickly collect millions of steps of interaction data for training. This high sample throughput is crucial for training a critic network that produces precise value estimates. As a result, traditional deep RL has demonstrated superhuman performance across various domains [2, 29, 32, 49]. In contrast, LLMs contain parameters on the scale of billions, and rollouting a single step often entails the autoregressive synthesis of hundreds or thousands of tokens. Consequently, the speed of online interaction and sample generation is prohibitively slow. This limitation leads to high variance in the value estimates produced by the critic network, frequently resulting in training instability. To address this challenge, we propose an MC-Critic designed to acquire low-variance value estimates in a rapid and cost-effective way.

MC-Critic Formulation Instead of training a parameterized critic network to approximate the value $V_{\pi_\theta}(s_t)$, MC-Critic is a parameter-free critic that estimates value directly via Monte-Carlo rollouts. Specifically, given a state s_t, the LLM agent interacts with the environment following the current policy π_θ to generate M trajectories $\{\tau_1, \cdots, \tau_M\}$, where each trajectory $\tau_i = \{s_t, c_t^i, r_t^i, s_{t+1}^i, c_{t+1}^i, r_{t+1}^i, \cdots, s_{t+T}^i\}$ extends for a maximum of T time steps. MC-Critic then calculates the value of state s_t as the average of the cumulative discounted rewards over these M trajectories, denoted as:

$$V_{\pi_\theta}^{\mathrm{MC}}(s_t) = \frac{1}{M}\sum_{i=1}^{M}\sum_{k=0}^{T-1}\gamma^k r_{k+t}^i. \tag{4}$$

This approach yields an unbiased value estimator, where variance can be reduced by increasing M. However, our empirical tests indicate that a 4B parameter LLM requires 3–6 seconds to generate a single-step reasoning chain and action. Consequently, utilizing the LLM to rollout M trajectories for estimating the value of a single state is prohibitively expensive and time-consuming. To mitigate this inference latency, we introduce a lightweight Monte-Carlo rollout scheme that employs a random policy π_{random} as a surrogate for the LLM policy π_θ to generate the M trajectories. We then derive the value estimate $V^{\text{MC}}_{\pi_{random}}(s_t)$ following Eq. (4). Although $V^{\text{MC}}_{\pi_{random}}(s_t)$ is theoretically suboptimal compared to $V^{\text{MC}}_{\pi_\theta}(s_t)$, it offers a low-variance value estimate very quickly and efficiently. For instance, in the game 2048, the random policy can rollout over 1,000 trajectories in less than 3 seconds.

Multi-Turn RL Optimization MC-Critic serves as a plug-and-play auxiliary value estimator, which is adaptable to any reinforcement learning algorithm requiring state value estimation. Below, we detail the integration of MC-Critic with two prevalent RL algorithms for LLMs: GRPO [28] and PPO [27], referring to the resulting variants as MC-GRPO and MC-PPO.

MC-GRPO The most straightforward way to extend GRPO to multi-turn scenarios is Trajectory-Level GRPO [36] (hereinafter referred to as Traj-GRPO). In Traj-GRPO, G trajectories $\{\tau_i\}_{i=1}^{G}$ are generated starting from a shared initial state s_0. For each trajectory $\tau_i = \{s_0, c_0^i, r_0^i, s_1^i, \cdots, s_{T_i}^i\}$, a trajectory-level reward $R(\tau_i) = \sum_{t=0}^{T_i-1} r_t^i$ is computed, where T_i represents the number of time steps in the trajectory τ_i. Subsequently, group normalization is applied to these trajectory-level rewards to derive trajectory-level advantages:

$$\hat{A}_i^{\text{Traj-GRPO}} = \frac{R(\tau_i) - \text{mean}(\{R(\tau_1), \cdots, R(\tau_G)\})}{\text{std}(\{R(\tau_1), \cdots, R(\tau_G)\})}. \tag{5}$$

To simplify the loss formulation, let us define the probability ratio $\rho_{t,j}^i(\theta)$ as:

$$\rho_{t,j}^i(\theta) = \frac{\pi_\theta(c_{t,j}^i | s_t^i, c_{t,<j}^i)}{\pi_{\theta_{\text{old}}}(c_{t,j}^i | s_t^i, c_{t,<j}^i)}, \tag{6}$$

where $c_{t,j}^i$ is the j^{th} token of c_t^i and $c_{t,<j}^i$ are the tokens before $c_{t,j}^i$. The loss function for Traj-GRPO is then defined as:

$$\begin{aligned}\mathcal{L}^{\text{Traj-GRPO}}(\theta) = -\frac{1}{\sum_{i=1}^{G}\sum_{t=0}^{T_i-1}|c_t^i|}\sum_{i=1}^{G}\sum_{t=0}^{T_i-1}\sum_{j=1}^{|c_t^i|}\min\Big[\rho_{t,j}^i(\theta)\hat{A}_i^{\text{Traj-GRPO}},\\ \text{clip}\left(\rho_{t,j}^i(\theta), 1-\epsilon, 1+\epsilon\right)\hat{A}_i^{\text{Traj-GRPO}}\Big],\end{aligned} \tag{7}$$

which simply assigns the shared trajectory-level advantage $\hat{A}_i^{\text{Traj-GRPO}}$ to all time steps outputs c_t^i within trajectory τ_i, lacking discriminative credit assignment for

different time steps. This can easily lead to model collapse [13]. Furthermore, it utilizes the trajectory-level reward and does not involve estimating the value of individual states, making it difficult to combine with MC-Critic.

To this end, we consider a step-level variant of GRPO for multi-turn scenarios, termed Step-GRPO. It rollouts a group of single-step samples instead of a group of trajectories. Specifically, we first rollout a trajectory and store all visited states in a state pool, which functions similarly to the question set in the math domain [28]. During each training step, we randomly select a batch of states $\{s_{t_u}\}_{u=1}^{b}$ from the state pool. For each state s_{t_u}, we generate G independent single-step samples $\{s_{t_u}, c_{t_u}^i, r_{t_u}^i\}_{i=1}^{G}$. Then, we compute the advantages for the single-step samples based on the step-level rewards $\{r_{t_u}^i\}_{i=1}^{G}$:

$$\hat{A}_{t_u,i}^{\text{Step-GRPO}} = \frac{r_{t_u}^i - \text{mean}(\{r_{t_u}^1, \cdots, r_{t_u}^G\})}{\text{std}(\{r_{t_u}^1, \cdots, r_{t_u}^G\})}. \tag{8}$$

The loss function for Step-GRPO is as follows:

$$\begin{aligned}\mathcal{L}^{\text{Step-GRPO}}(\theta) = -\frac{1}{\sum_{u=1}^{b}\sum_{i=1}^{G}|c_{t_u}^i|}\sum_{u=1}^{b}\sum_{i=1}^{G}\sum_{j=1}^{|c_{t_u}^i|}\min\Big[\rho_{t_u,j}^i(\theta)\hat{A}_{t_u,i}^{\text{Step-GRPO}},\\ \text{clip}\left(\rho_{t_u,j}^i(\theta), 1-\epsilon, 1+\epsilon\right)\hat{A}_{t_u,i}^{\text{Step-GRPO}}\Big].\end{aligned} \tag{9}$$

Building upon Step-GRPO, we introduce MC-GRPO, which augments Step-GRPO with the MC-Critic. The primary distinction lies in the substitution of the immediate step-reward with an action-value function to compute the advantage for the single-step sample $\{s_{t_u}, c_{t_u}^i, r_{t_u}^i\}$. Specifically, the action-value function for $\{s_{t_u}, c_{t_u}^i, r_{t_u}^i\}$ (i.e., $\{s_{t_u}, z_{t_u}^i, a_{t_u}^i, r_{t_u}^i\}$) is estimated via the MC-Critic as follows:

$$Q_{\pi_{random}}^{\text{MC}}(s_{t_u}, a_{t_u}^i) = \mathbb{E}_{s_{t_u+1}\sim\mathcal{P}(\cdot|s_{t_u},a_{t_u}^i)}\left[r_{t_u}^i + \gamma V_{\pi_{random}}^{\text{MC}}(s_{t_u+1})\right]. \tag{10}$$

Subsequently, we apply group normalization to derive the advantage:

$$\hat{A}_{t_u,i}^{\text{MC-GRPO-relative}} = \frac{Q_{\pi_{random}}^{\text{MC}}(s_{t_u}, a_{t_u}^i) - \text{mean}(\{Q_{\pi_{random}}^{\text{MC}}(s_{t_u}, a_{t_u}^i)\}_{i=1}^{G})}{\text{std}(\{Q_{\pi_{random}}^{\text{MC}}(s_{t_u}, a_{t_u}^i)\}_{i=1}^{G})}. \tag{11}$$

In practice, we found that if the actions $\{a_{t_u}^i\}_{i=1}^{G}$ selected within a group are identical, their corresponding action-values are also identical. This results in an advantage of zero for the entire group (i.e., zero policy gradients), making it difficult to improve the policy for that state s_{t_u}. Instead of the common practice of discarding these samples through dynamic sampling [50], we aim to leverage these group samples to further enhance the policy. Specifically, when $\{a_{t_u}^i\}_{i=1}^{G}$ are identical, we replace the group-relative baseline with the average of action-values of all actions in the action space $\mathcal{A}$ (i.e., an absolute baseline) to compute the advantage:

$$\hat{A}_{t_u,i}^{\text{MC-GRPO-absolute}} = \frac{Q_{\pi_{random}}^{\text{MC}}(s_{t_u}, a_{t_u}^i) - \text{mean}(\{Q_{\pi_{random}}^{\text{MC}}(s_{t_u}, a)\}_{a\in\mathcal{A}})}{\text{std}(\{Q_{\pi_{random}}^{\text{MC}}(s_{t_u}, a)\}_{a\in\mathcal{A}})}. \tag{12}$$

The final advantage formulation for MC-GRPO is defined as:

$$\hat{A}^{\text{MC-GRPO}}_{t_u,i} = \begin{cases} \hat{A}^{\text{MC-GRPO-relative}}_{t_u,i} & \text{if } \{a^i_{t_u}\}_{i=1}^G \text{ are not identical,} \\ \hat{A}^{\text{MC-GRPO-absolute}}_{t_u,i} & \text{if } \{a^i_{t_u}\}_{i=1}^G \text{ are identical.} \end{cases}$$

This yields the loss function for MC-GRPO:

$$\mathcal{L}^{\text{MC-GRPO}}(\theta) = -\frac{1}{\sum_{u=1}^{b}\sum_{i=1}^{G}|c^i_{t_u}|}\sum_{u=1}^{b}\sum_{i=1}^{G}\sum_{j=1}^{|c^i_{t_u}|}\min\Big[\rho^i_{t_u,j}(\theta)\hat{A}^{\text{MC-GRPO}}_{t_u,i}, \\ \text{clip}\left(\rho^i_{t_u,j}(\theta), 1-\epsilon, 1+\epsilon\right)\hat{A}^{\text{MC-GRPO}}_{t_u,i}\Big]. \tag{13}$$

The only difference between $\mathcal{L}^{\text{MC-GRPO}}$ and $\mathcal{L}^{\text{Step-GRPO}}$ is in the calculation of the advantage. By incorporating the MC-Critic, the LLM agent is incentivized to prioritize long-term returns over immediate single-step gains, thereby fostering multi-turn lookahead reasoning capabilities.

MC-PPO Recent advancements have extended PPO to multi-turn scenarios, such as VAGEN [34] and Turn-PPO [13]. These methods typically concatenate the entire interaction history as context before the agent makes a decision at each time step. However, in multi-turn scenarios where a single trajectory contains hundreds of time steps, this practice often causes the context length to exceed the maximum token limit (e.g., 32,768 tokens). To address this, we adopt a simplified context formulation where the LLM agent relies solely on the current state for decision-making, discarding historical interaction information. Building upon this setting, we propose Step-PPO, a PPO variant tailored for multi-turn scenarios.

Similar to Turn-PPO [13], Step-PPO employs a turn-level critic V_ϕ to compute the turn-level generalized advantage estimation (GAE) [26] advantage for policy updates, where all tokens within a turn share the same turn-level advantage. Specifically, we first rollout a trajectory $\tau = (s_0, c_0, r_0, s_1, c_1, r_1, \ldots, s_T)$ and designate the critic's output at the final token of the response c_t as the turn-level value $V_\phi(s_t, c_t)$. We then calculate the advantage for each turn using GAE:

$$\delta^{\text{Step-PPO}}_t = r_t + \gamma V_\phi(s_{t+1}, c_{t+1}) - V_\phi(s_t, c_t), \tag{14}$$

$$\hat{A}^{\text{Step-PPO}}_t = \sum_{l=t}^{T-1}(\gamma\lambda)^{l-t}\delta^{\text{Step-PPO}}_l, \tag{15}$$

Subsequently, the policy loss for Step-PPO is defined as:

$$\mathcal{L}^{\text{Step-PPO}}(\theta) = -\frac{1}{\sum_{t=0}^{T-1}|c_t|}\sum_{t=0}^{T-1}\sum_{j=1}^{|c_t|}\min\Big[\rho_{t,j}(\theta)\hat{A}^{\text{Step-PPO}}_t, \\ \text{clip}\left(\rho_{t,j}(\theta), 1-\epsilon, 1+\epsilon\right)\hat{A}^{\text{Step-PPO}}_t\Big], \tag{16}$$

and the value loss is given by:

$$\mathcal{L}^{\text{Step-PPO}}(\phi) = \frac{1}{T}\sum_{t=0}^{T-1}\left(V_\phi(s_t, c_t) - R_t\right)^2, \text{ where } R_t = \hat{A}_t^{\text{Step-PPO}} + V_{\phi_{\text{old}}}(s_t, c_t). \tag{17}$$

As previously discussed, the inherent sample inefficiency of LLMs results in high variance in the values estimated by the trained critic V_ϕ. To mitigate this, we propose MC-PPO, which leverages the MC-Critic to assist in estimating a turn-level value with reduced variance. Concretely, we compute a weighted sum of the value $V^{\text{MC}}_{\pi_{random}}(s_t)$ estimated by the MC-Critic and the value $V_\phi(s_t, c_t)$ output by the turn-level critic to derive the step-level value for MC-PPO:

$$V^{\text{MC-PPO}}(s_t) = (1-\omega)V_\phi(s_t, c_t) + \omega V^{\text{MC}}_{\pi_{random}}(s_t), \tag{18}$$

where $\omega \in [0, 1]$ represents the weight of the value estimated by MC-Critic. We then use this new value to calculate the advantage and update the policy and the critic, identical to the process in Step-PPO.

4 Experiments

We evaluate ProAct on two representative long-horizon decision-making benchmarks: *2048*, a stochastic environment that requires planning under uncertainty and contains hundreds of turns in a single trajectory, and *Sokoban*, a deterministic planning task with shorter turns per trajectory but sparse rewards. These environments pose complementary challenges for lookahead reasoning and error accumulation. To evaluate generalization beyond the training distribution, we further test ProAct on multiple environment variants.

All experiments use *Qwen3-4B-Instruct-2507* [43] as the backbone. We first perform SFT on our GLAD dataset (25K and 8K trajectories for 2048 and Sokoban), followed by RL enhanced with our MC-Critic. Evaluation metrics are cumulative tile merge score for 2048 and average successful box pushes for Sokoban. Further details regarding the environments and hyperparameters are deferred to Appendix A–C.

4.1 Results

SFT with GLAD As shown in Tab. 1, our 4B GLAD-trained model consistently outperforms all open-source and several strong closed-source baselines. Furthermore, GLAD demonstrates strong out-of-distribution (OOD) robustness, achieving consistent gains on 2048 variants (3×3, 3072) and Sokoban variants (unseen levels, modified action/symbol spaces). This confirms that distilled lookahead reasoning provides a substantial margin across diverse scenarios.

By observing specific cases, we found that the model's reasoning behavior improves significantly after GLAD supervision. Analysis of a representative 2048 task reveals that while the base model often produces verbose but unstable reasoning—characterized by redundant steps and occasional hallucinations of

Table 1: Performance comparison on 2048 and Sokoban under multiple environment variants. For 2048, **4×4** denotes the standard 4×4 grid setting, **3×3** corresponds to a reduced grid-size variant, and **3072** indicates a modified environment where the minimum tile value is 3. For Sokoban, **Unseen** evaluates performance on held-out levels that do not appear in the SFT training data, **Action** modifies the action space, and **Symbol** alters the symbolic representation of the map. Columns marked with * indicate evaluation on environment variants that are strictly unseen during both SFT and RL training phases. For brevity, Qwen3 baselines are denoted by parameter count; all use the Instruct-2507 release (i.e., Qwen3-235B-A22B-Instruct-2507, Qwen3-30B-A3B-Instruct-2507, Qwen3-4B-Instruct-2507).

	2048			**Sokoban**		
Model	**4×4**	**3×3***	**3072***	**Unseen**	**Action***	**Symbol***
Closed-source Models						
GPT-5 [24]	4040.0	**1184.0**	**6962.0**	**1.89**	**1.83**	1.94
Claude-4.5-Sonnet [1]	166.7	109.3	120.0	1.06	0.78	1.33
Doubao-Seed-1.6 [3]	1877.3	300.0	2054.0	1.22	0.61	1.56
Doubao-Seed-1.8 [4]	**4662.7**	545.3	4210.0	1.80	1.44	**2.00**
UI-TARS-1.5 [25]	2616.0	466.7	3920.0	0.44	0.44	0.39
Open-source Models						
Qwen3-235B-A22B	634.7	274.7	2688.0	0.61	0.33	0.56
Qwen3-30B-A3B	838.7	230.7	1740.0	0.44	0.39	0.50
Qwen3-4B (Base Model)	721.3	187.3	1603.0	0.39	0.44	0.56
Base Model + GLAD	3335.3	429.2	4565.7	0.72	0.52	0.67
Base Model + GLAD + MC-Critic	**4503.8**	**464.4**	**6013.7**	**0.94**	**0.60**	**0.70**

board states—the GLAD-supervised model generates a much more compact and accurate intermediate analysis. Specifically, the supervised model demonstrates an ability to perform explicit lookahead, simulating various action outcomes and systematically comparing tile-merging potential before selecting the optimal move. A detailed presentation of these cases can be found in Appendix E.

RL with MC-Critic We evaluate the effectiveness of MC-Critic in two complementary reinforcement learning settings: (1) *RL fine-tuning on top of GLAD-initialized models*, and (2) *training from scratch starting from the base instruction-tuned model*. Besides, we analyze two important hyperparameters used in MC-Critic and summarize the recipe of MC-Critic.

Trained from GLAD SFT Checkpoints. We first investigate MC-Critic as an RL refinement applied on top of GLAD-initialized agents. Starting from the GLAD SFT checkpoint, we fine-tune the model using PPO and GRPO, both with and without MC-Critic, to isolate its contribution.

As shown in Fig. 3(a-d), MC-Critic consistently improves performance on both 2048 and Sokoban. Notably, the incorporation of MC-Critic benefits both GRPO and PPO, leading to higher final scores. Moreover, as reported in Tab. 2,

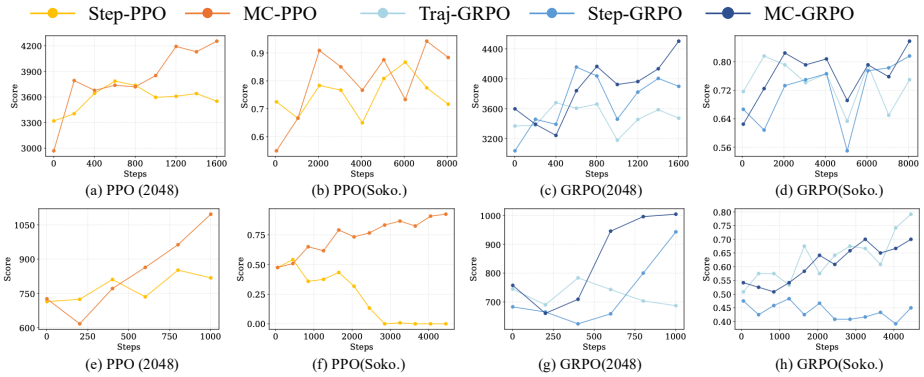

Fig. 3: Performance of RL methods and MC-Critic. **Top Row (a-d):** Models initialized from GLAD SFT checkpoints. **Bottom Row (e-h):** Models trained from scratch. MC-Critic consistently enhances both PPO and GRPO across diverse environments and initialization strategies.

these improvements are not limited to the training distribution; performance gains consistently transfer to unseen variants.

Overall, these results demonstrate that, when initialized from a strong GLAD prior, MC-Critic effectively exploits Monte-Carlo returns to further refine long-horizon decision-making, while maintaining robust generalization to OOD settings.

Train From Scratch. We next evaluate whether MC-Critic can enhance RL performance without GLAD supervision, training all methods directly from the base Qwen3-4B-Instruct-2507 model. Fig. 3(e-h) reports results on 2048 under the standard setting and Sokoban on the training levels. Across both environments, MC-PPO achieves the highest scores, confirming the effectiveness of integrating MC-Critic into policy optimization. In 2048, MC-GRPO consistently outperforms other GRPO-based variants, further supporting the benefits of our approach. In Sokoban, MC-GRPO performs comparably to Traj-GRPO. This can be attributed to the simplicity of the training Sokoban levels: most levels are solved within a few steps, making entire trajectories sufficient as single training samples, without incurring significant variance. However, as trajectory length increases, reward variance accumulates, destabilizing training for Traj-GRPO. This effect is evident in 2048, where Traj-GRPO performance degrades, whereas MC-GRPO remains stable and continues to improve, highlighting the robustness of MC-Critic in long-horizon tasks.

To evaluate generalization, we further test all methods on various environment variants. Tab. 2 shows that MC-Critic outperforms the baseline methods on both 2048 and Sokoban variants, demonstrating its superior robustness and generalization.

Table 2: Performance of RL methods with and without MC-Critic on environment variants.

	2048		Sokoban		
Method	**3×3**	**3072**	**Unseen-RL**	**Action**	**Symbol**
Trained from GLAD SFT Checkpoints					
Traj-GRPO	459.0	5457.0	1.10	0.60	0.64
Step-GRPO	457.0	5820.3	1.00	0.56	0.63
Step-PPO	**487.3**	6248.7	0.90	**0.62**	0.65
MC-GRPO	464.4	**6013.7**	1.05	0.60	**0.70**
MC-PPO	465.0	5818.1	**1.18**	**0.62**	0.68
Trained From Scratch					
Traj-GRPO	202.0	1760.7	0.50	0.73	0.90
Step-GRPO	188.0	1792.9	0.25	0.53	0.86
Step-PPO	202.7	1912.9	0.45	0.55	0.88
MC-GRPO	194.2	1754.7	**0.55**	0.80	**1.03**
MC-PPO	**239.8**	**2229.1**	0.53	**0.96**	0.98

Analysis. Finally, we analyze the impact of two critical hyperparameters within MC-Critic, as previously defined in Eq. (4): the number of trajectories, denoted by M, and the maximum number of steps per trajectory, denoted by T. We present the detailed experimental results and analysis in the Appendix D, from which we derive the MC-Critic recipe: For environments with dense rewards (e.g., 2048), M should be set as large as possible, provided that the interaction efficiency allows. For environments with sparse rewards (e.g., Sokoban), M should not be excessive, as a smaller M may yield better results. Additionally, T should be set to the average number of steps required for a successful trajectory and should not be overly large.

5 Conclusion

In this work, we present ProAct, a comprehensive framework that enhances LLM agents with accurate lookahead and stable policy optimization in long-horizon interactive environments. To mitigate compounding simulation errors and high-variance value estimation, ProAct adopts a two-stage training paradigm. First, Grounded LookAhead Distillation (GLAD) distills MCTS trajectories into concise, environment-grounded reasoning chains, bridging expensive inference-time search and efficient policy learning. Second, the Monte-Carlo Critic (MC-Critic) provides a low-variance, plug-and-play value estimator that improves the stability of multi-turn agentic RL training. Experiments on stochastic (2048) and deterministic (Sokoban) benchmarks show that a 4B model trained with ProAct outperforms open-source baselines and achieves generalization comparable to state-of-the-art proprietary models.

References

1. Anthropic: Introducing claude sonnet 4.5 (2025)
2. Berner, C., Brockman, G., Chan, B., Cheung, V., Dębiak, P., Dennison, C., Farhi, D., Fischer, Q., Hashme, S., Hesse, C., et al.: Dota 2 with large scale deep reinforcement learning. arXiv preprint arXiv:1912.06680 (2019)
3. Bytedance Seed Team: Seed1.6 (2025), `https://seed.bytedance.com/en/seed1_6`
4. Bytedance Seed Team: Seed1.8 (2025), `https://seed.bytedance.com/en/seed1_8`
5. Cao, S., Li, D., Zhao, F., Yuan, S., Hegde, S.R., Chen, C., Ruan, C., Griggs, T., Liu, S., Tang, E., et al.: Skyrl-agent: Efficient rl training for multi-turn llm agent. arXiv preprint arXiv:2511.16108 (2025)
6. Guo, D., Yang, D., Zhang, H., Song, J., Zhang, R., Xu, R., Zhu, Q., Ma, S., Wang, P., Bi, X., et al.: Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. arXiv preprint arXiv:2501.12948 (2025)
7. Hao, S., Gu, Y., Ma, H., Hong, J., Wang, Z., Wang, D., Hu, Z.: Reasoning with language model is planning with world model. In: Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing. pp. 8154–8173 (2023)
8. He, L., Ye, D., Tan, J., Wang, X., Shen, L.: Robust policy expansion for offline-to-online rl under diverse data corruption. Advances in Neural Information Processing Systems **38**, 149262–149289 (2026)
9. Hsieh, C.Y., Li, C.L., Yeh, C.K., Nakhost, H., Fujii, Y., Ratner, A., Krishna, R., Lee, C.Y., Pfister, T.: Distilling step-by-step! outperforming larger language models with less training data and smaller model sizes. In: Findings of the Association for Computational Linguistics: ACL 2023. pp. 8003–8017 (2023)
10. Huang, J., Lyu, Z., Zhang, F., Deng, Y., Ye, D., An, B.: Uncertainty-aware tree search for efficient llm reasoning (2026)
11. Jaech, A., Kalai, A., Lerer, A., Richardson, A., El-Kishky, A., Low, A., Helyar, A., Madry, A., Beutel, A., Carney, A., et al.: Openai o1 system card. arXiv preprint arXiv:2412.16720 (2024)
12. Jiang, T., Lin, Z., Li, L., Li, Y.C., Guan, C., Yuan, L., Zhang, Z., Yu, Y., Ye, D.: Multi-agent in-context coordination via decentralized memory retrieval. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 40, pp. 22363–22371 (2026)
13. Li, J., Zhou, P., Meng, R., Vadera, M.P., Li, L., Li, Y.: Turn-ppo: Turn-level advantage estimation with ppo for improved multi-turn rl in agentic llms. arXiv preprint arXiv:2512.17008 (2025)
14. Li, J., Zhang, Q., Yu, Y., Fu, Q., Ye, D.: More agents is all you need. arXiv preprint arXiv:2402.05120 (2024)
15. Li, P., Hu, Z., Shang, Z., Wu, J., Liu, Y., Liu, H., Gao, Z., Shi, C., Zhang, B., Zhang, Z., et al.: Efficient multi-turn rl for gui agents via decoupled training and adaptive data curation. arXiv preprint arXiv:2509.23866 (2025)
16. Li, Z.Z., Zhang, D., Zhang, M.L., Zhang, J., Liu, Z., Yao, Y., Xu, H., Zheng, J., Wang, P.J., Chen, X., et al.: From system 1 to system 2: A survey of reasoning large language models. arXiv preprint arXiv:2502.17419 (2025)
17. Lin, X., Ning, Y., Zhang, J., Dong, Y., Liu, Y., Wu, Y., Qi, X., Sun, N., Shang, Y., Wang, K., et al.: Llm-based agents suffer from hallucinations: A survey of taxonomy, methods, and directions. arXiv preprint arXiv:2509.18970 (2025)
18. Lin, Z., Liu, F., Yang, Y., Lyu, J., Gao, Y., Liu, Y., Lu, Z., Yu, Y., Yang, M., Li, J., Ye, D., Jiang, J.: Ui-voyager: A self-evolving gui agent learning via failed experience. arXiv preprint arXiv:2603.24533 (2026)

19. Lu, Z., Lin, Z., Jia, W., Tian, C., Ye, D., Li, P., Jin, L., Liu, N., Xu, G., Feng, W.: Hisr: Hindsight information modulated segmental process rewards for multi-turn agentic reinforcement learning. arXiv preprint arXiv:2603.18683 (2026)
20. Luo, J., Zhang, W., Yuan, Y., Zhao, Y., Yang, J., Gu, Y., Wu, B., Chen, B., Qiao, Z., Long, Q., et al.: Large language model agent: A survey on methodology, applications and challenges. arXiv preprint arXiv:2503.21460 (2025)
21. Lyu, J., Lin, Z., Fujimoto, S., Yang, K., Chen, Y., Yang, S., Lu, Z., Ye, D.: Debiased model-based representations for sample-efficient continuous control. arXiv preprint arXiv:2605.11711 (2026)
22. Lyu, J., Yan, M., Qiao, Z., Liu, R., Ma, X., Ye, D., Yang, J.W., Lu, Z., Li, X.: Cross-domain offline policy adaptation with optimal transport and dataset constraint. In: The Thirteenth International Conference on Learning Representations (2025)
23. Lyu, J., Yang, J., Qiao, Z., Liu, R., Liu, Z., Ye, D., Lu, Z., Li, X.: Temporal difference learning with constrained initial representations. Information Sciences p. 123774 (2026)
24. OpenAI: Introducing gpt-5 (2025)
25. Qin, Y., Ye, Y., Fang, J., Wang, H., Liang, S., Tian, S., Zhang, J., Li, J., Li, Y., Huang, S., et al.: Ui-tars: Pioneering automated gui interaction with native agents. arXiv preprint arXiv:2501.12326 (2025)
26. Schulman, J., Moritz, P., Levine, S., Jordan, M., Abbeel, P.: High-dimensional continuous control using generalized advantage estimation. arXiv preprint arXiv:1506.02438 (2015)
27. Schulman, J., Wolski, F., Dhariwal, P., Radford, A., Klimov, O.: Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347 (2017)
28. Shao, Z., Wang, P., Zhu, Q., Xu, R., Song, J., Bi, X., Zhang, H., Zhang, M., Li, Y., Wu, Y., et al.: Deepseekmath: Pushing the limits of mathematical reasoning in open language models. arXiv preprint arXiv:2402.03300 (2024)
29. Silver, D., Huang, A., Maddison, C.J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., et al.: Mastering the game of go with deep neural networks and tree search. nature **529**(7587), 484–489 (2016)
30. Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., Lanctot, M., Sifre, L., Kumaran, D., Graepel, T., et al.: Mastering chess and shogi by self-play with a general reinforcement learning algorithm. arXiv preprint arXiv:1712.01815 (2017)
31. Tian, C., Lu, Z., Liu, H., Wang, X., Li, S., Chen, Y., Lv, W., Lin, Z., Diao, J., Ye, D.: Faithful-mr1: Faithful multimodal reasoning via anchoring and reinforcing visual attention. arXiv preprint arXiv:2605.22072 (2026)
32. Vinyals, O., Babuschkin, I., Czarnecki, W.M., Mathieu, M., Dudzik, A., Chung, J., Choi, D.H., Powell, R., Ewalds, T., Georgiev, P., et al.: Grandmaster level in starcraft ii using multi-agent reinforcement learning. nature **575**(7782), 350–354 (2019)
33. Wang, H., Zou, H., Song, H., Feng, J., Fang, J., Lu, J., Liu, L., Luo, Q., Liang, S., Huang, S., et al.: Ui-tars-2 technical report: Advancing gui agent with multi-turn reinforcement learning. arXiv preprint arXiv:2509.02544 (2025)
34. Wang, K., Zhang, P., Wang, Z., Gao, Y., Li, L., Wang, Q., Chen, H., Wan, C., Lu, Y., Yang, Z., et al.: Vagen: Reinforcing world model reasoning for multi-turn vlm agents. arXiv preprint arXiv:2510.16907 (2025)
35. Wang, L., Ma, C., Feng, X., Zhang, Z., Yang, H., Zhang, J., Chen, Z., Tang, J., Chen, X., Lin, Y., et al.: A survey on large language model based autonomous agents. Frontiers of Computer Science **18**(6), 186345 (2024)

36. Wang, Z., Wang, K., Wang, Q., Zhang, P., Li, L., Yang, Z., Jin, X., Yu, K., Nguyen, M.N., Liu, L., et al.: Ragen: Understanding self-evolution in llm agents via multi-turn reinforcement learning. arXiv preprint arXiv:2504.20073 (2025)
37. Wei, J., Wang, X., Schuurmans, D., Bosma, M., Xia, F., Chi, E., Le, Q.V., Zhou, D., et al.: Chain-of-thought prompting elicits reasoning in large language models. Advances in neural information processing systems **35**, 24824–24837 (2022)
38. Wei, T., Yang, Y., Xing, J., Shi, Y., Lu, Z., Ye, D.: Gtr: Guided thought reinforcement prevents thought collapse in rl-based vlm agent training. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 18855–18865 (2025)
39. Wu, F., Chen, C., Tan, Z., Zhang, T., Xu, X., Qian, Y., Gao, D., Zhu, L., Zhu, Q., Tan, Y., Ji, D., Lin, G., Chen, T., Ye, D., Liu, F.: Claw ai lab: An autonomous multi-agent research team. arXiv preprint arXiv:2605.22662 (2026)
40. Xi, Z., Huang, J., Liao, C., Huang, B., Guo, H., Liu, J., Zheng, R., Ye, J., Zhang, J., Chen, W., et al.: Agentgym-rl: Training llm agents for long-horizon decision making through multi-turn reinforcement learning. arXiv preprint arXiv:2509.08755 (2025)
41. Xie, Z., Hu, Z., Ye, F., Zhang, X., Chai, H., Liu, Z., Wu, P., Zhang, G., Liao, Y., Hu, X., Ye, D., Miao, C., Yan, S.: Pask: Toward intent-aware proactive agents with long-term memory. arXiv preprint arXiv:2604.08000 (2026)
42. Yan, M., Lyu, J., Sun, S., Qiao, Z., Yang, J., Lin, Z., Ye, D., Li, X.: Cross-domain offline policy adaptation via selective transition correction. arXiv preprint arXiv:2602.05776 (2026)
43. Yang, A., Li, A., Yang, B., Zhang, B., Hui, B., Zheng, B., Yu, B., Gao, C., Huang, C., Lv, C., et al.: Qwen3 technical report. arXiv preprint arXiv:2505.09388 (2025)
44. Yang, K., Xu, X., Chen, Y., Liu, W., Lyu, J., Lin, Z., Ye, D., Yang, S.: Entropic: Towards stable long-term training of llms via entropy stabilization with proportional-integral control. arXiv preprint arXiv:2511.15248 (2025)
45. Yang, M., Yang, Y., Lu, Z., Zhou, W., Li, H.: Hierarchical multi-agent skill discovery. Advances in Neural Information Processing Systems **36**, 61759–61776 (2023)
46. Yang, M., Zhao, J., Hu, X., Zhou, W., Zhu, J., Li, H.: Ldsa: Learning dynamic subtask assignment in cooperative multi-agent reinforcement learning. Advances in neural information processing systems **35**, 1698–1710 (2022)
47. Yao, S., Yu, D., Zhao, J., Shafran, I., Griffiths, T., Cao, Y., Narasimhan, K.: Tree of thoughts: Deliberate problem solving with large language models. Advances in neural information processing systems **36**, 11809–11822 (2023)
48. Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K.R., Cao, Y.: React: Synergizing reasoning and acting in language models. In: The eleventh international conference on learning representations (2022)
49. Ye, D., Chen, G., Zhang, W., Chen, S., Yuan, B., Liu, B., Chen, J., Liu, Z., Qiu, F., Yu, H., et al.: Towards playing full moba games with deep reinforcement learning. Advances in Neural Information Processing Systems **33**, 621–632 (2020)
50. Yu, Q., Zhang, Z., Zhu, R., Yuan, Y., Zuo, X., Yue, Y., Dai, W., Fan, T., Liu, G., Liu, L., et al.: Dapo: An open-source llm reinforcement learning system at scale. arXiv preprint arXiv:2503.14476 (2025)
51. Yu, Y., Zhang, Q., Li, J., Fu, Q., Ye, D.: Affordable generative agents. arXiv preprint arXiv:2402.02053 (2024)
52. Zelikman, E., Wu, Y., Mu, J., Goodman, N.: Star: Bootstrapping reasoning with reasoning. Advances in Neural Information Processing Systems **35**, 15476–15488 (2022)

53. Zhang, F., Li, J., Li, Y.C., Zhang, Z., Yu, Y., Ye, D.: Improving sample efficiency of reinforcement learning with background knowledge from large language models. IEEE Transactions on Neural Networks and Learning Systems (2025)
54. Zhang, X., Chan, H., Ye, D., Cai, Y., Zhao, M.: Ad hoc teamwork via offline goal-based decision transformers. In: Forty-second International Conference on Machine Learning (2025)
55. Zhao, H., Yang, Y., Lin, Z., Ye, D., Miao, C.: Hiro-nav: Hybrid reasoning enables efficient embodied navigation. arXiv preprint arXiv:2604.08232 (2026)
56. Zhao, J., Yang, M., Zhao, Y., Hu, X., Zhou, W., Zhu, J., Li, H.: Mcmarl: Parameterizing value function via mixture of categorical distributions for multi-agent reinforcement learning. arXiv preprint arXiv:2202.10134 (2022)
57. Zhao, J., Zhao, Y., Wang, W., Yang, M., Hu, X., Zhou, W., Hao, J., Li, H.: Coach-assisted multi-agent reinforcement learning framework for unexpected crashed agents. Frontiers of Information Technology & Electronic Engineering **23**(7), 1032–1042 (2022)
58. Zheng, H., Shen, L., Luo, Y., Ye, D., Du, B., Shen, J., Tao, D.: Decision mixer: Integrating long-term and local dependencies via dynamic token selection for decision-making. In: Forty-second International Conference on Machine Learning (2025)
59. Zheng, H., Shen, L., Luo, Y., Ye, D., Xu, S., Du, B., Shen, J., Tao, D.: Value-guided decision transformer: A unified reinforcement learning framework for online and offline settings. Advances in Neural Information Processing Systems **38**, 71655–71684 (2026)
60. Zhou, S., Zhou, T., Yang, Y., Long, G., Ye, D., Jiang, J., Zhang, C.: Wall-e: World alignment by neurosymbolic learning improves world model-based llm agents. Advances in Neural Information Processing Systems **38**, 65352–65391 (2026)
61. Zhou, Y., Jiang, S., Tian, Y., Weston, J., Levine, S., Sukhbaatar, S., Li, X.: Sweet-rl: Training multi-turn llm agents on collaborative reasoning tasks. arXiv preprint arXiv:2503.15478 (2025)
62. Zhou, Y., Zanette, A., Pan, J., Levine, S., Kumar, A.: Archer: Training language model agents via hierarchical multi-turn rl. arXiv preprint arXiv:2402.19446 (2024)