SAPO: SINGLE-ROLLOUT AUTOREGRESSIVE POLICY OPTIMIZATION FOR AGENTIC REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Agentic reinforcement learning (RL) has emerged as an important post-training approach for enhancing the capabilities of Large Language Models (LLMs). However, existing methods face a trade-off between policy performance and resource efficiency. Conventional Proximal Policy Optimization (PPO) implementations incur substantial memory overhead from a separate critic, whereas critic-free group-relative methods require multiple rollouts and face potential learning bottlenecks on long-horizon tasks. In this work, we propose **S**ingle-rollout **A**utoregressive **P**olicy **O**ptimization (**SAPO**), an efficient PPO-style framework that unifies policy optimization and value learning within a single causal language model. SAPO exploits the autoregressive structure of LLMs to sequentially generate action and value estimation at distinct causal boundaries with shared parameters, and then jointly optimizes the PPO objectives and an auxiliary on-policy SARSA objective with turn-level generalized advantage estimation, where the latter is designed to facilitate value learning. Extensive experiments on ALFWorld and WebShop with Qwen2.5-1.5B/7B and Qwen3-14B demonstrate that SAPO reduces peak GPU memory usage by **23.1%** and per-iteration runtime by **24.8%** over strong PPO baseline, while matching or slightly improving task success rate. Our experiments also show that SAPO outperforms Group Relative Policy Optimization (GRPO) and recent cutting-edge variants in both task success and training stability.

1 INTRODUCTION

Recent advances in agentic reinforcement learning (RL) have transformed large language models from passive text generators into agents capable of long-horizon reasoning, tool use, web search, and code execution. Large-scale RL post-training has produced substantial improvements in mathematical reasoning and coding Shao et al. (2024); Guo et al. (2025); Hu et al. (2025), while interactive systems such as Search-R1 show that policies can learn to acquire external information through multi-turn interaction Jin et al. (2025). These successes suggest that learning directly from environmental feedback is becoming a central path toward more autonomous and general-purpose language agents.

Most agentic RL methods build on proximal policy optimization (PPO) or its critic-free, group-relative variants. PPO provides token-level or turn-level credit assignment through a learned value function and generalized advantage estimation (GAE), but typically requires a separate, often policy-scale critic with substantial memory and computation costs Ouyang et al. (2022); Stiennon et al. (2020). GRPO removes this critic by normalizing rewards among multiple trajectories sampled for the same task Shao et al. (2024). This efficiency comes with three limitations. When trajectories in a group receive identical or nearly identical rewards, which is common under sparse feedback and long-horizon failures, the normalized advantages vanish and produce zero gradients Yu et al. (2025). Moreover, assigning one trajectory-level advantage uniformly to all generated tokens provides weak temporal credit assignment and cannot generalize value estimates across states, prompts, or policy iterations. Learned critics can identify low-value repetitive or erroneous prefixes that group-relative estimators miss Yuan et al. (2025); Yue et al. (2025); Hu et al. (2025). Finally, group-relative estimation creates an unfavorable tradeoff between computation and estimation quality. Small groups yield noisy baselines, while larger groups improve stability only by multiplying rollout cost and

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

imposing synchronization across long trajectories of varying lengths Choi et al. (2026); Hou et al. (2026).

Several methods attempt to mitigate these problems. DAPO improves group-relative learning through dynamic sampling, asymmetric clipping, and token-level policy losses, but remains dependent on multiple rollouts and group statistics Yu et al. (2025). ReMax and RLOO replace the critic with greedy or leave-one-out baselines, while VinePPO estimates intermediate values through Monte Carlo continuations. These approaches reduce training-state memory but retain coarse credit assignment or introduce additional generation cost Li et al. (2024); Ahmadian et al. (2024); Kazemnejad et al. (2024). Conversely, Open-Reasoner-Zero, VC-PPO, and VAPO restore explicit value models and show that a well-trained critic can improve long-horizon reasoning through stronger temporal credit assignment Hu et al. (2025); Yuan et al. (2025); Yue et al. (2025). They nevertheless retain a separate, often policy-scale value network. Hydra-PPO shares a frozen backbone but requires role-specific adapters and heads, and its fully shared variant reveals potential interference between policy and value learning Santacroce et al. (2023). POISE reuses actor hidden states through a lightweight probe but depends on cross-rollout estimation Choi et al. (2026), while SAO supports asynchronous single-rollout training with a separately pretrained critic and more frequent value updates Hou et al. (2026). Efficient single-rollout learning with explicit temporal value estimation therefore remains unresolved.

We address this gap with Single-Rollout Autoregressive Policy Optimization (SAPO), a framework that represents the policy, state value, and action value within one causal language model. The core insight is that the information requirements of value learning align with two natural boundaries in autoregressive generation. The context before an action supports the state value $V(s)$, while the extended context after that action supports the action value $Q(s, a)$. SAPO uses a shared two-token value basis to read both quantities from the language model logits at these boundaries, with causal masking preserving their distinct conditioning and the value tokens excluded from the action distribution. From one sampled trajectory, SAPO computes turn-specific advantages and value targets with GAE, then jointly optimizes the clipped policy loss, a state value objective, and an auxiliary on-policy SARSA objective for the action value in one shared model update. This design removes the independent critic forward and backward path, its optimizer state, and the repeated sequence processing required by a second model. It also removes the multiple synchronized rollouts needed for group-relative estimation. SAPO therefore turns the causal structure of language generation into an architectural principle for efficient value-based RL, retaining explicit temporal credit assignment within the resource profile of a single model and a single rollout.

We evaluate SAPO on ALFWorld and WebShop using Qwen backbones from 1.5B to 14B under matched optimization settings. Across model scales, SAPO delivers PPO-level task success without a separate critic model and remains competitive with recent fine-grained credit-assignment methods, without requiring group sampling or additional dense reward signals. It consistently outperforms GRPO and exhibits more stable learning, supporting the value of explicit temporal credit assignment in long-horizon interaction. The unified design also reduces peak GPU memory by 15.2%–35.6% across the evaluated settings and shortens each measured training iteration by 22.6%–23.6% relative to PPO. SAPO thus provides a low-cost, single-rollout realization of PPO-style training that preserves learned values and temporal credit assignment. Our contributions are threefold:

- We introduce a single-rollout autoregressive policy optimization method that unifies policy optimization and value learning within a single causal language model.
- We systematically explore and incorporate mechanisms such as on-policy SARSA, regularized value learning spaces, and dual logits to enhance the effectiveness of SAPO.
- We demonstrate that SAPO achieves a favorable combination of powerful performance, training stability, memory efficiency, and runtime efficiency.

2 Related Work

Agentic RL extends reinforcement learning with verifiable rewards from single-turn response optimization to multi-turn decision making, enabling LLM agents to learn from the consequences of actions taken in external environments. A prominent line of work builds on critic-free, group-relative optimization. GRPO Shao et al. (2024) replaces a learned value baseline with within-prompt out-

come comparison, a formulation used by DeepSeek-R1 Guo et al. (2025) and refined by DAPO Yu et al. (2025) with dynamic sampling and asymmetric clipping, GSPO Zheng et al. (2025a) with sequence-level importance ratios, and GVPO Zhang et al. (2025) through variance-aware gradient weighting. This paradigm has been extended to interactive settings: Search-R1 Jin et al. (2025) and DeepResearcher Zheng et al. (2025b) optimize iterative retrieval, ToRL Li et al. (2025b) and WebSailor Li et al. (2025a) train tool use and web navigation, and RAGEN Wang et al. (2025) characterizes optimization instabilities in multi-turn training. Long-horizon tasks have also motivated more structured credit mechanisms. ARChER Zhou et al. (2024) and HiPER Peng et al. (2026) decompose planning and execution across temporal scales, AgentGym-RL Xi et al. (2026) progressively increases the interaction horizon, and IGPO Wang et al. (2026a) derives dense intrinsic rewards from information gain during search.

Efficiency-oriented LLM RL has developed along critic-free advantage estimation, parameter sharing between policy and value functions, and systems-level execution. ReMax Li et al. (2024) and RLOO Ahmadian et al. (2024) replace learned values with greedy or leave-one-out baselines, while GRPO Shao et al. (2024) normalizes rewards within each prompt group. Eliminating the value network reduces parameters and optimizer state, but these estimators require additional decoded trajectories and provide no value generalization across states. VinePPO Kazemnejad et al. (2024) obtains intermediate value estimates from branched Monte Carlo continuations, likewise exchanging critic training for increased rollout computation. Parameter-sharing approaches retain explicit value learning while reducing model duplication. Hydra-PPO Santacroce et al. (2023) places policy and value adapters over a frozen backbone; J-Hydra Santacroce et al. (2023) further shares the trainable adapter, with potential interference between the two objectives. POISE Choi et al. (2026) predicts returns from actor hidden states using a lightweight probe, but relies on cross-rollout construction to decouple value estimation from the evaluated trajectory. SAO Hou et al. (2026) addresses rollout underutilization through asynchronous single-rollout collection, while retaining a separately pretrained critic with more frequent value updates. HybridFlow Sheng et al. (2025) instead improves distributed execution through model placement, resharding, and scheduling, independently of the advantage estimator. SAPO is distinguished by jointly representing the policy, state value, and action value in one causal autoregressive model, thereby eliminating both group sampling and the separate critic backbone while retaining explicit temporal credit assignment.

3 PRELIMINARIES

Multi-Turn Agentic RL. Multi-turn agentic RL trains an LLM policy through sequential interactions with an environment. We model each interaction episode as a finite-horizon Markov decision process (MDP), conditioned on a task instance $\boldsymbol{x} \sim p(X)$. At turn t, the policy samples a textual action $\boldsymbol{a}_t \sim \pi_\theta(\cdot \mid \boldsymbol{s}_t)$ from the current state $\boldsymbol{s}_t \in \mathcal{S}$. The environment then executes the action, returns a reward $r_t \in \mathbb{R}$, and transitions to $\boldsymbol{s}_{t+1}$. An episode with T interaction turns produces the trajectory $\boldsymbol{\tau} = \big((\boldsymbol{s}_t, \boldsymbol{a}_t, r_t)\big)_{t=1}^{T}$. In our interaction format for tasks such as ALFWorld, each response comprises a reasoning segment enclosed in `<think>...</think>` and an executable command enclosed in `<action>...</action>`, only the command is passed to the environment. Feedback in these settings is often sparse and delayed, with an outcome reward $R(\boldsymbol{\tau}) \in [0, R_{max}]$ assigned upon episode completion.

3.1 GROUP-RELATIVE POLICY OPTIMIZATION

Recent agentic RL methods for LLMs often follow a group-relative policy optimization framework. For a task instance $\boldsymbol{x}$, the old policy $\pi_{\theta_{\text{old}}}$ generates a group of N complete rollouts, denoted by $\mathcal{G}_x = \{\boldsymbol{\tau}_i\}_{i=1}^{N}$. Each trajectory receives a scalar reward $R(\boldsymbol{\tau}_i)$ that measures task success or outcome quality. Whereas PPO typically uses a learned value critic to estimate advantages Schulman et al. (2017), group-relative methods derive advantage estimates directly from the rewards of rollouts sampled for the same task:

$$A(\boldsymbol{\tau}_i) = \left[\texttt{GroupNormalization}\left(\{R(\boldsymbol{\tau}_j)\}_{j=1}^{N}\right)\right]_i .$$

In GRPO Shao et al. (2024), this estimate is obtained by subtracting the group mean from each trajectory reward and dividing by the group standard deviation. By using within-group reward statistics as the baseline, GRPO eliminates the need for a separate value critic, reducing the associated memory and computational costs.

3.2 Proximal Policy Optimization

Proximal Policy Optimization (PPO) is an actor–critic reinforcement learning algorithm widely used for LLM post-training Schulman et al. (2017). Given prompts sampled from a dataset, the policy generates response trajectories that are evaluated by task-specific environments or learned reward models. The policy is optimized using a clipped surrogate objective:

$$\mathcal{L}_{\text{PPO}} = \mathbb{E}_t \left[\min \left(\rho_t \hat{A}_t,\ \text{clip}(\rho_t, 1-\epsilon, 1+\epsilon) \hat{A}_t \right) \right],$$

where $\rho_t = \pi_\theta(a_t \mid s_t) / \pi_{\theta_{\text{old}}}(a_t \mid s_t)$ is the likelihood ratio between the current and old policies, $\hat{A}_t$ is the estimated advantage, and ϵ controls the clipping range. Clipping limits the incentive for large policy updates, helping stabilize optimization. A learned value critic $V_\phi(s_t)$ estimates expected future returns and supports advantage estimation, commonly through generalized advantage estimation. These estimates provide learning signals for earlier decisions when rewards are delayed. In token-level LLM optimization, states represent token prefixes or interaction histories, and actions correspond to generated tokens. Although the critic supports credit assignment across long trajectories, maintaining and training it introduces additional memory and computational costs. Practical LLM post-training pipelines may also retain a reference policy and an old-policy snapshot for regularization and rollout collection, further increasing resource requirements.

4 Method

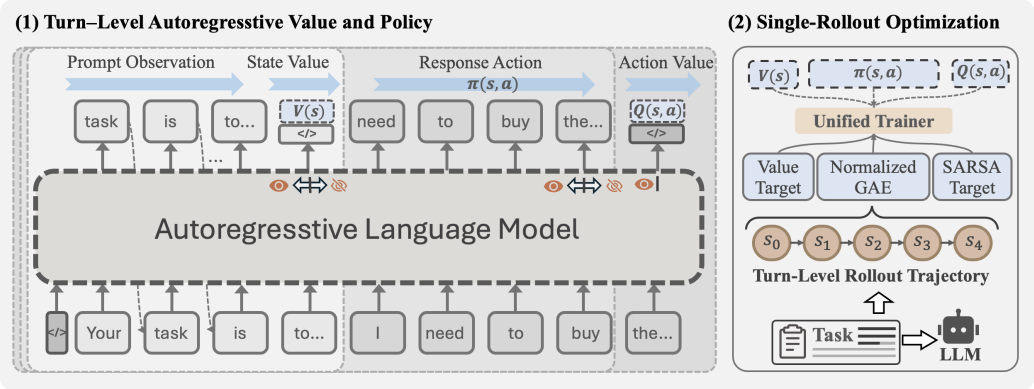

Figure 1: **Overview of SAPO. Left**: Value and policy autoregressive processes. A shared autoregressive language model generates actions and reads V and Q from logits at the pre- and post-action boundaries. **Right**: Loss structure and training pipeline. One rollout per task supplies turn-level value V targets, on-policy SARSA Q target and advantages, which are broadcast to response tokens for policy optimization.

We propose Single-rollout Autoregressive Policy Optimization (SAPO), an efficient PPO-style framework that unifies policy optimization and value learning within a single causal language model. SAPO represents the policy, state value, and action value within one causal language model (Figure 1). It reads values at the boundaries before and after each generated response, constructs turn-level advantages and value targets from one rollout per task, and jointly optimizes token-level policy and turn-level value objectives. Algorithm 1 in Appendix A summarizes the complete training procedure.

4.1 Autoregressive Policy and Value Representation

Causal Conditioning at Action Boundaries. At turn t, let $\boldsymbol{c}_t^s$ represent the current interaction history and $\boldsymbol{a}_t = (a_{t,1}, \ldots, a_{t,M_t})$ the generated response. The state value conditions on $\boldsymbol{c}_t^s$, while the action value additionally conditions on the complete response, using $\boldsymbol{c}_t^{sa} = [\boldsymbol{c}_t^s; \boldsymbol{a}_t]$. We read

these values at the final state-context position and the final response position, respectively. Causal masking prevents the state-value readout from accessing the current action and the action-value readout from accessing the subsequent reward or observation.

Shared Scalar Readout. Let $\boldsymbol{z}_\theta(\boldsymbol{c}) \in \mathbb{R}^{|\mathcal{W}|}$ denote the next-token logits after context $\boldsymbol{c}$. We reserve two vocabulary entries, w^+ and w^-, and use their logit difference as a scalar value readout:

$$v_\theta(\boldsymbol{c}) = \frac{z_{\theta,w^+}(\boldsymbol{c}) - z_{\theta,w^-}(\boldsymbol{c})}{\tau_v}, \qquad v_\theta(\boldsymbol{c}) \in \mathbb{R}, \tag{1}$$

where $\tau_v > 0$ controls the readout scale. The readout is unconstrained: we apply neither output clipping nor a probability mapping. Given a return-scale factor $R_{\max} > 0$, the two boundaries yield

$$V_\theta(\boldsymbol{s}_t) = R_{\max} v_\theta(\boldsymbol{c}_t^s), \qquad Q_\theta(\boldsymbol{s}_t, \boldsymbol{a}_t) = R_{\max} v_\theta(\boldsymbol{c}_t^{sa}). \tag{2}$$

Value-Target Scaling. We train the readout against value targets divided by $R_{\max}$ (Section 4.3), while using the rescaled V and Q in Equation 2 for temporal-difference and advantage computation. This separates the numerical scale of value regression from the reward scale used for policy learning. Empirically, with outcome rewards scaled to 10, regressing scaled targets improves performance over directly fitting unscaled targets. We hypothesize that requiring large logit differences for value prediction perturbs the pretrained model's logit structure and increases interference with policy learning. Target scaling encourages smaller readouts without imposing a hard output range.

Action Distribution and Shared Evaluation. The reserved entries serve only as readouts and are never sampled or appended to the context. We exclude them from the action vocabulary, $\mathcal{W}_{\text{act}} = \mathcal{W} \setminus \{w^+, w^-\}$, and define

$$\pi_\theta(a \mid \boldsymbol{c}) = \frac{\exp z_{\theta,a}(\boldsymbol{c})}{\sum_{w \in \mathcal{W}_{\text{act}}} \exp z_{\theta,w}(\boldsymbol{c})}, \qquad a \in \mathcal{W}_{\text{act}}. \tag{3}$$

We use this support for rollout sampling, policy likelihoods, and regularization. The actor evaluation on each sampled state–response sequence supplies both token log probabilities and the two value readouts. During optimization, these predictions likewise share one forward pass, eliminating the separate critic's forward/backward computation and training state. Policy and value losses still interact through the shared parameters.

4.2 Turn-Level Advantages and Value Targets

For each task, we sample one trajectory $\boldsymbol{\tau} = ((\boldsymbol{s}_t, \boldsymbol{a}_t, r_t, d_t))_{t=1}^T$ with the rollout policy $\pi_{\theta_{\text{old}}}$, where d_t indicates episode termination. All targets use frozen rollout-time predictions, expressed on the original reward scale through Equation 2, and are held fixed during optimization.

Turn-Level Generalized Advantage Estimation. With $V_t^{\text{old}} = V_{\theta_{\text{old}}}(\boldsymbol{s}_t)$, we apply GAE along the interaction turns:

$$\delta_t = r_t + \gamma(1 - d_t) V_{t+1}^{\text{old}} - V_t^{\text{old}}, \tag{4}$$

$$A_t^{\text{GAE}} = \delta_t + \gamma\lambda(1 - d_t) A_{t+1}^{\text{GAE}}, \tag{5}$$

with $A_{T+1}^{\text{GAE}} = 0$. The task horizon defines episode termination in our finite-horizon setting, so the final bootstrap value is zero. This recursion propagates delayed rewards across turns, with λ controlling the bias–variance trade-off.

State- and Action-Value Targets. We supervise V with a λ-return and Q with an on-policy SARSA target:

$$y_t^V = V_t^{\text{old}} + A_t^{\text{GAE}}, \qquad y_t^Q = r_t + \gamma(1 - d_t) Q_{t+1}^{\text{old}}, \tag{6}$$

where $Q_{t+1}^{\text{old}} = Q_{\theta_{\text{old}}}(\boldsymbol{s}_{t+1}, \boldsymbol{a}_{t+1})$ uses the next action from the same rollout; the bootstrap term is zero at termination. The state value supplies the baseline for GAE, while Q provides auxiliary supervision after each response. This on-policy SARSA update adds negligible overhead, and its purpose is to strengthen state-value learning. Because V and Q use the same two reserved logit coordinates, gradients from post-action Q regression shape the shared value representation used to produce the pre-action V estimate. Table 2 empirically supports this coupling, as removing $\mathcal{L}_Q$ consistently degrades performance. Policy advantages still use V-based GAE rather than $Q - V$, so action-value errors do not directly enter the advantage estimates.

Advantage Normalization. For environments with an invalid-action indicator m_t^{inv}, we retain the adjustment $\widetilde{A}_t = A_t^{\text{GAE}} - c_{\text{inv}} m_t^{\text{inv}}$; otherwise $c_{\text{inv}} = 0$. We compute the mean $\mu_{\mathcal{B}}$ and standard deviation $\sigma_{\mathcal{B}}$ of $\widetilde{A}_t$ over all valid turns in the rollout batch, and set $\widehat{A}_t = \frac{\widetilde{A}_t - \mu_{\mathcal{B}}}{\sigma_{\mathcal{B}} + \epsilon_{\text{adv}}}$. Statistics are computed before broadcasting advantages to response tokens, so each turn contributes once regardless of response length. This standardization pools turns across trajectories without requiring multiple rollouts of the same task.

4.3 Joint Policy and Value Optimization

SAPO applies two forms of supervision to the shared language model. The policy objective is evaluated at every generated token, whereas the value objectives supervise one pre-action and one post-action readout per interaction turn. All advantages and value targets are computed from the rollout policy and detached before optimization.

Token-Level Clipped Policy Objective. Consider a rollout batch $\mathcal{B} = \{\boldsymbol{\tau}_i\}_{i=1}^{B}$. Let $\mathcal{I}_{\mathcal{B}} = \{(i,t,j) : 1 \le i \le B,\, 1 \le t \le T_i,\, 1 \le j \le M_{i,t}\}$ index its valid generated response tokens, and let $N_{\text{tok}} = |\mathcal{I}_{\mathcal{B}}|$. For token $a_{i,t,j}$, the causal context is $\boldsymbol{c}_{i,t,j} = [\boldsymbol{c}^s_{i,t}; \boldsymbol{a}_{i,t,<j}]$. The standardized turn advantage is broadcast to every token in the corresponding response, $\widehat{A}_{i,t,j} = \text{sg}(\widehat{A}_{i,t})$, where sg denotes stop-gradient. We minimize the standard PPO clipped surrogate over generated tokens:

$$\mathcal{L}_{\text{clip}}(\theta) = -\frac{1}{N_{\text{tok}}} \sum_{(i,t,j)\in\mathcal{I}_{\mathcal{B}}} \min\Bigg(\frac{\pi_\theta(a_{i,t,j} \mid \boldsymbol{c}_{i,t,j})}{\pi_{\theta_{\text{old}}}(a_{i,t,j} \mid \boldsymbol{c}_{i,t,j})} \widehat{A}_{i,t,j}, \\ \text{clip}\Big(\frac{\pi_\theta(a_{i,t,j} \mid \boldsymbol{c}_{i,t,j})}{\pi_{\theta_{\text{old}}}(a_{i,t,j} \mid \boldsymbol{c}_{i,t,j})}, 1-\epsilon, 1+\epsilon \Big) \widehat{A}_{i,t,j} \Bigg). \tag{7}$$

The token mask includes all generated response tokens and excludes prompt, observation, and padding tokens. Hence, SAPO estimates credit at the turn level but applies the likelihood ratio, clipping, and gradient update at the token level.

Scaled Value Regression. Let $\mathcal{T}_{\mathcal{B}} = \{(i,t) : 1 \le i \le B,\, 1 \le t \le T_i\}$ denote the valid turns and $N_{\text{turn}} = |\mathcal{T}_{\mathcal{B}}|$. For $X \in \{V, Q\}$, we define

$$v^V_{i,t} = v_\theta(\boldsymbol{c}^s_{i,t}), \qquad v^Q_{i,t} = v_\theta(\boldsymbol{c}^{sa}_{i,t}), \qquad \bar{y}^X_{i,t} = \frac{\text{sg}(y^X_{i,t})}{R_{\max}}. \tag{8}$$

Thus, the readouts are optimized on the scaled target space, while Equation 2 restores the reward scale when constructing temporal-difference targets and advantages. Following PPO, we limit changes in the normalized value prediction relative to its rollout-time value:

$$v^{X,\text{clip}}_{i,t} = v^{X,\text{old}}_{i,t} + \text{clip}\Big(v^X_{i,t} - v^{X,\text{old}}_{i,t}, -\epsilon_X, \epsilon_X \Big). \tag{9}$$

The corresponding turn-level regression loss is

$$\mathcal{L}_X(\theta) = \frac{1}{2N_{\text{turn}}} \sum_{(i,t)\in\mathcal{T}_{\mathcal{B}}} \max\Big(\big(v^X_{i,t} - \bar{y}^X_{i,t}\big)^2, \Big(v^{X,\text{clip}}_{i,t} - \bar{y}^X_{i,t}\Big)^2 \Big), \qquad X \in \{V, Q\}. \tag{10}$$

This clipping constrains the update in the regression objective; it does not bound the scalar readout itself. Averaging over turns also prevents response length from changing the relative weight of value supervision.

Unified Objective. The final loss combines policy optimization, the two causal value objectives, reference-policy regularization, and entropy regularization:

$$\mathcal{L}_{SAPO}(\theta) = \mathcal{L}_{\text{clip}} + c_V \mathcal{L}_V + c_Q \mathcal{L}_Q + \beta \mathcal{L}_{\text{KL}} - c_H \mathcal{H}(\pi_\theta). \tag{11}$$

The policy, state-value, and action-value terms use separate masks but update the same transformer and language-model head in one backward pass. Coefficients c_V and c_Q control interference between generation and value learning, while the KL and entropy terms regularize policy drift and exploration. This objective retains distinct supervision for the three roles without allocating role-specific model parameters.

5 EXPERIMENTS

We evaluate SAPO on two multi-turn environments and organize our experiments around four research questions: (1) *As a low-cost PPO-style training paradigm, can SAPO match or surpass conventional PPO in task performance and training stability?* (2) *How much does SAPO reduce the peak memory usage and runtime overhead of PPO?* (3) *How does SAPO compare with SAPO and recent strong baselines?* (4) *How much does each component contribute, practically how effectively does SAPO learn value estimates?*

5.1 EXPERIMENTAL SETUP

Benchmarks and Baselines. We train and evaluate SAPO on two challenging multi-turn benchmarks, ALFWorld Shridhar et al. (2020) and WebShop Yao et al. (2022a). ALFWorld evaluates long-horizon textual reasoning and decision-making in an embodied household environment, with 3,827 task instances spanning six categories. WebShop evaluates interactive web navigation and product search over nearly 1.1 million products and 12,000 user instructions. We compare SAPO with baselines covering complementary agent and optimization paradigms. Prompting: GPT-4o Achiam et al. (2023) and Gemini-2.5-Pro Team et al. (2023), together with ReAct Yao et al. (2022b) and Reflexion Shinn et al. (2023). RL training: PPO Schulman et al. (2017), GRPO Shao et al. (2024), RLOO Ahmadian et al. (2024), together with recent methods with dense signals: GiGPO Feng et al. (2026), EMPG Wang et al. (2026b), and GraphGPO Cheng et al. (2026). Further descriptions of these baselines are provided in Appendix C.

Type	Method	ALFWorld							WebShop	
		Pick	Look	Clean	Heat	Cool	Pick2	All	Score	Succ.
Closed-Source Model										
Prompting	GPT-4o	75.3	60.8	31.2	56.7	21.6	49.8	48.0	31.8	23.7
Prompting	Gemini-2.5-Pro	92.8	63.3	62.1	69.0	26.6	58.7	60.3	42.5	35.9
Qwen2.5-1.5B-Instruct										
Prompting	ReAct	17.4	20.5	15.7	6.2	7.7	2.0	12.8	40.1	11.3
Prompting	Reflexion	35.3	22.2	21.7	13.6	19.4	3.7	21.8	55.8	21.9
RL Training	RLOO	$88.3_{\pm 3.0}$	$52.8_{\pm 8.6}$	$71.0_{\pm 5.9}$	$62.8_{\pm 8.7}$	$66.4_{\pm 5.5}$	$56.9_{\pm 4.7}$	$69.7_{\pm 2.5}$	$73.9_{\pm 5.6}$	$52.1_{\pm 6.7}$
RL Training	EMPG	85.5	33.5	78.9	76.2	74.7	89.1	73.7	80.4	60.8
RL Training	$\text{GiGPO}_{\text{w/ std}}$	$94.4_{\pm 5.9}$	$67.5_{\pm 4.6}$	$94.8_{\pm 3.8}$	$94.4_{\pm 7.8}$	$79.8_{\pm 4.7}$	$76.4_{\pm 5.4}$	$86.7_{\pm 1.7}$	$83.1_{\pm 1.6}$	$65.0_{\pm 3.2}$
RL Training	$\text{GiGPO}_{\text{w/o std}}$	$96.0_{\pm 1.4}$	$76.5_{\pm 3.9}$	$91.8_{\pm 5.5}$	$91.3_{\pm 6.3}$	$71.7_{\pm 8.4}$	$79.5_{\pm 7.7}$	$86.1_{\pm 4.7}$	$83.5_{\pm 1.8}$	$67.4_{\pm 4.5}$
RL Training	GraphGPO	$95.2_{\pm 1.6}$	$\mathbf{100.0}_{\pm 0.0}$	$85.3_{\pm 2.6}$	$85.7_{\pm 5.83}$	$96.3_{\pm 2.6}$	$93.7_{\pm 2.24}$	$92.7_{\pm 1.3}$	$\mathbf{89.3}_{\pm 1.5}$	$\mathbf{78.7}_{\pm 3.9}$
RL Training	GRPO	$85.3_{\pm 1.5}$	$53.7_{\pm 8.0}$	$84.5_{\pm 6.8}$	$78.2_{\pm 7.9}$	$59.7_{\pm 5.0}$	$53.5_{\pm 5.6}$	$72.8_{\pm 3.6}$	$75.8_{\pm 3.5}$	$56.8_{\pm 3.8}$
RL Training	PPO	$\mathbf{100.0}_{\pm 0.0}$	$79.5_{\pm 4.2}$	$97.9_{\pm 2.5}$	$\mathbf{100.0}_{\pm 0.0}$	$81.4_{\pm 1.8}$	$\mathbf{94.5}_{\pm 4.2}$	$93.0_{\pm 1.1}$	$82.7_{\pm 1.2}$	$65.6_{\pm 2.7}$
RL Training	**SAPO**	$96.4_{\pm 1.4}$	$81.2_{\pm 5.1}$	$\mathbf{100.0}_{\pm 0.0}$	$\mathbf{100.0}_{\pm 0.0}$	$\mathbf{97.2}_{\pm 3.2}$	$87.7_{\pm 5.1}$	$\mathbf{94.6}_{\pm 1.4}$	$80.1_{\pm 2.1}$	$66.0_{\pm 1.5}$
Qwen2.5-7B-Instruct										
Prompting	ReAct	48.5	35.4	34.3	13.2	18.2	17.6	31.2	46.2	19.5
Prompting	Reflexion	62.0	41.6	44.9	30.9	36.3	23.8	42.7	58.1	28.8
RL Training	RLOO	$87.6_{\pm 4.3}$	$78.2_{\pm 8.3}$	$87.3_{\pm 5.8}$	$81.3_{\pm 7.6}$	$71.9_{\pm 5.2}$	$48.9_{\pm 8.4}$	$75.5_{\pm 4.6}$	$80.3_{\pm 3.2}$	$65.7_{\pm 4.0}$
RL Training	EMPG	92.9	75.2	74.8	86.3	73.7	65.3	78.5	81.0	69.3
RL Training	$\text{GiGPO}_{\text{w/ std}}$	$97.7_{\pm 1.6}$	$82.7_{\pm 7.9}$	$98.8_{\pm 1.6}$	$83.7_{\pm 7.2}$	$89.3_{\pm 8.2}$	$79.2_{\pm 6.6}$	$90.8_{\pm 1.3}$	$84.4_{\pm 2.9}$	$72.8_{\pm 3.2}$
RL Training	$\text{GiGPO}_{\text{w/o std}}$	$91.8_{\pm 5.4}$	$88.6_{\pm 6.3}$	$95.9_{\pm 3.2}$	$90.2_{\pm 2.6}$	$86.5_{\pm 5.5}$	$85.2_{\pm 7.5}$	$90.2_{\pm 2.3}$	$86.2_{\pm 2.6}$	$75.2_{\pm 3.8}$
RL Training	GraphGPO	$\mathbf{100.0}_{\pm 0.0}$	$\mathbf{100.0}_{\pm 0.0}$	$91.4_{\pm 1.5}$	$92.9_{\pm 5.8}$	$94.4_{\pm 0.0}$	$92.1_{\pm 2.2}$	$95.3_{\pm 1.1}$	$86.9_{\pm 0.7}$	$80.3_{\pm 1.3}$
RL Training	GRPO	$90.8_{\pm 5.1}$	$66.1_{\pm 6.7}$	$89.3_{\pm 5.4}$	$74.7_{\pm 6.9}$	$72.5_{\pm 5.4}$	$64.7_{\pm 7.3}$	$77.6_{\pm 5.2}$	$79.3_{\pm 2.8}$	$66.1_{\pm 3.7}$
RL Training	PPO	$\mathbf{100.0}_{\pm 0.0}$	$\mathbf{100.0}_{\pm 0.0}$	$97.9_{\pm 3.6}$	$88.4_{\pm 5.3}$	$89.7_{\pm 5.8}$	$\mathbf{98.7}_{\pm 2.2}$	$95.8_{\pm 1.2}$	$87.7_{\pm 1.8}$	$82.3_{\pm 2.2}$
RL Training	**SAPO**	$98.8_{\pm 1.6}$	$85.5_{\pm 11.2}$	$\mathbf{99.1}_{\pm 2.0}$	$\mathbf{100.0}_{\pm 0.0}$	$\mathbf{95.6}_{\pm 4.7}$	$92.3_{\pm 8.2}$	$\mathbf{96.1}_{\pm 1.1}$	$\mathbf{88.4}_{\pm 1.7}$	$\mathbf{83.6}_{\pm 1.9}$
Qwen3-14B-Instruct										
RL Training	GRPO	$99.0_{\pm 1.8}$	$69.1_{\pm 5.8}$	$85.9_{\pm 5.0}$	$79.7_{\pm 12.8}$	$75.6_{\pm 24.5}$	$79.7_{\pm 13.6}$	$83.8_{\pm 8.6}$	$85.0_{\pm 6.1}$	$77.2_{\pm 6.6}$
RL Training	PPO	$\mathbf{100.0}_{\pm 0.0}$	$93.6_{\pm 6.3}$	$\mathbf{100.0}_{\pm 0.0}$	$86.7_{\pm 23.1}$	$\mathbf{100.0}_{\pm 0.0}$	$\mathbf{100.0}_{\pm 0.0}$	$97.9_{\pm 2.3}$	$\mathbf{92.4}_{\pm 1.5}$	$\mathbf{84.8}_{\pm 1.8}$
RL Training	**SAPO**	$99.0_{\pm 1.8}$	$\mathbf{95.8}_{\pm 7.2}$	$\mathbf{100.0}_{\pm 0.0}$	$\mathbf{97.6}_{\pm 4.1}$	$98.7_{\pm 2.2}$	$\mathbf{100.0}_{\pm 0.0}$	$\mathbf{98.7}_{\pm 0.5}$	$90.0_{\pm 1.3}$	$84.4_{\pm 0.8}$

Table 1: Performance results on ALFWorld and WebShop. For each RL training method, we report the mean and standard deviation over three random seeds. The ALFWorld contains six categories: Pick & Place (Pick), Examine in Light (Look), Clean & Place (Clean), Heat & Place (Heat), Cool & Place (Cool), and Pick Two & Place (Pick2). The baseline $\text{GiGPO}_{\text{w/o std}}$ replaces the group std with one Feng et al. (2026).

Implementation Details. We use Qwen2.5-1.5B-Instruct, Qwen2.5-7B-Instruct, and Qwen3-14B-Instruct base models. For fair comparisons against the core PPO and GRPO baselines, we follow the matched protocol of Feng et al. Feng et al. (2026) implemented in the verl-agent framework. Notably, **PPO employs turn-level GAE consistented with SAPO, rather than the token-level GAE reported in GiGPO** Feng et al. (2026). All methods use a learning rate of 1×10^{-6}, a KL

regularization coefficient of 0.01, and 150 optimization steps. We cap each episode at 50 turns for ALFWorld and 15 for WebShop. Training is performed using NVIDIA H200, H20, and A40 GPUs. Full hyperparameters and details are provided in the Appendix B.

5.2 Main Results

ALFWorld and WebShop Results. Table 1 shows that SAPO achieves PPO-level performance across both benchmarks and all model scales while substantially outperforming GRPO. On ALFWorld, SAPO reaches aggregate success rates of 94.6%, 96.1%, and 98.7% at 1.5B, 7B, and 14B, slightly exceeding PPO at each scale and surpassing GRPO by 14.9–21.8 points. On WebShop, SAPO remains within 1.3 points of PPO in success rate but improves over GRPO by 7.2–17.5 points. This result is notable because PPO remains a strong optimization baseline, consistently outperforming GRPO under the matched protocol across both environments and all model scales. By avoiding group-relative estimates, it is less susceptible to weak temporal credit assignment, unstable optimization, and vanishing advantages when within-group rewards are identical.

Compared with methods that construct dense credit signals from sparse outcomes, SAPO requires no step-level supervision. It achieves the best aggregate ALFWorld result at 1.5B, although GraphGPO remains stronger on WebShop. At 7B, SAPO leads all dense-signal baselines, reaching 96.1% ALFWorld success, an 88.4 WebShop score, and 83.6% WebShop success. SAPO shows lower standard deviations in most multi-run comparisons. At 14B, its standard deviation is 0.5 versus 8.6 for GRPO on ALFWorld and 0.8 versus 6.6 on WebShop success. Overall, SAPO retains PPO's value learning and temporal credit assignment without a separate critic or multi-rollout group sampling, delivering PPO-level optimization quality at lower cost.

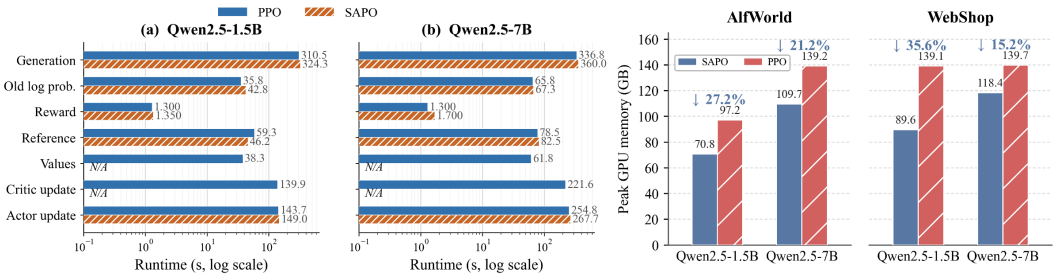

Figure 2: **Computational resource comparison between SAPO and PPO** using Qwen2.5-1.5B and Qwen2.5-7B on ALFWorld and WebShop under matched experimental settings. The left panel reports the per-iteration runtime breakdown across training stages in seconds on a logarithmic scale, where N/A denotes stages not used by SAPO. The right panel reports peak GPU memory usage in GB during training on NVIDIA H200 GPUs; the 1.5B experiments use one GPU and the 7B experiments use two GPUs.

Computational Resource Comparison. Figure 2 compares the training cost of SAPO and PPO under matched settings. SAPO consistently lowers peak GPU memory across both environments and model scales, with reductions ranging from 15.2% to 35.6%. The largest reduction occurs on WebShop with Qwen2.5-1.5B, where peak memory drops from 139.1 GB to 89.6 GB; the reduction persists at 7B and on ALFWorld. These savings arise from SAPO's unified actor–critic design, which performs value learning within the policy model rather than maintaining a separate critic and its optimizer state.

The runtime comparison on ALFWorld reflects both this architectural simplification and SAPO's autoregressive rollout design. Relative to PPO, SAPO reduces the measured per-iteration time by 22.6% at 1.5B and 23.6% at 7B. Beyond removing direct critic computation, each autoregressive rollout supplies both policy-gradient and value-learning signals through causal-boundary readouts, allowing the two objectives to be optimized jointly in one shared-model update. SAPO therefore avoids repeated sequence processing through a second model and the independent critic optimization that together consume roughly one quarter of a PPO iteration at both scales. The shared update makes SAPO's actor-update stage only modestly slower, leaving a substantial net reduction in runtime.

Table 2: Ablation study comparing variants without action value learning loss ($-\mathcal{L}_{\mathcal{Q}}$) and advantage normalization ($-Norm$). The best performance under the same setting is highlighted in bold. We report the average success rate (%) from three seeds.

Method	ALFWorld						
	Pick	Clean	Cool	Look	Heat	Pick2	All
SAPO (1.5B)	$96.4_{\pm 1.4}$	$81.2_{\pm 5.1}$	$\mathbf{100.0}_{\pm 0.0}$	$\mathbf{100.0}_{\pm 0.0}$	$\mathbf{97.2}_{\pm 3.2}$	$\mathbf{87.7}_{\pm 5.1}$	$\mathbf{94.6}_{\pm 1.4}$
$-\mathcal{L}_{\mathcal{Q}}$ (1.5B)	$\mathbf{100.0}_{\pm 0.0}$	$81.6_{\pm 13.3}$	$98.6_{\pm 2.4}$	$\mathbf{100.0}_{\pm 0.0}$	$85.3_{\pm 9.4}$	$84.3_{\pm 5.6}$	$92.2_{\pm 0.8}$
$-Norm$ (1.5B)	$97.1_{\pm 2.7}$	$\mathbf{84.1}_{\pm 14.1}$	$91.7_{\pm 10.3}$	$96.7_{\pm 4.5}$	$84.8_{\pm 7.4}$	$83.6_{\pm 8.6}$	$89.1_{\pm 3.4}$
SAPO (7B)	$98.8_{\pm 1.6}$	$85.5_{\pm 11.2}$	$99.1_{\pm 2.0}$	$\mathbf{100.0}_{\pm 0.0}$	$\mathbf{95.6}_{\pm 4.7}$	$\mathbf{92.3}_{\pm 8.2}$	$\mathbf{96.1}_{\pm 1.1}$
$-\mathcal{L}_{\mathcal{Q}}$ (7B)	$95.9_{\pm 4.4}$	$82.7_{\pm 4.3}$	$96.2_{\pm 6.6}$	$97.4_{\pm 4.4}$	$82.3_{\pm 5.4}$	$87.4_{\pm 11.0}$	$91.2_{\pm 2.0}$
$-Norm$ (7B)	$\mathbf{100.0}_{\pm 0.0}$	$\mathbf{94.7}_{\pm 7.3}$	$\mathbf{100.0}_{\pm 0.0}$	$98.3_{\pm 3.3}$	$91.4_{\pm 3.5}$	$89.0_{\pm 2.0}$	$95.5_{\pm 2.0}$

5.3 Ablation Experiments and Value Analysis

Ablation Study. Table 2 examines the auxiliary action-value loss and advantage normalization on ALFWorld. Removing $\mathcal{L}_Q$ reduces aggregate success from 94.6% to 92.2% at 1.5B and from 96.1% to 91.2% at 7B. This consistent drop shows that SARSA-style action-value supervision improves temporal credit assignment beyond state-value learning alone. Advantage normalization has a scale-dependent effect. Removing it lowers 1.5B success to 89.1% and increases the standard deviation from 1.4 to 3.4, whereas the 7B model remains competitive at 95.5% but is less stable across runs. Overall, action-value learning provides consistent gains across model scales, while advantage normalization is especially important for the smaller model and stable optimization.

Learned Value Correlations.

Figure 3 tracks the Pearson correlations between the learned value predictions and their single-trajectory Monte Carlo returns throughout training. Both correlations rise from near zero to approximately 0.8, with most of the improvement occurring within the first 50 steps and a stable plateau thereafter. The state value V improves faster early in training, consistent with its λ-return supervision, while the action value Q increases more gradually under the one-step SARSA target. Their sustained correlations show that the two causal-boundary readouts learn its return aligned value estimates despite sharing a single language model. In particular, V provides an informative state-dependent baseline for GAE, while Q captures action-conditioned outcome information. These results support SAPO's central premise that explicit temporal value learning can be retained without a separate critic or additional rollouts.

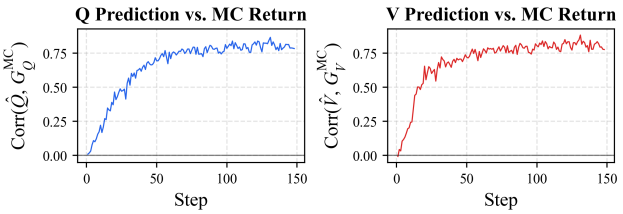

Figure 3: **Pearson correlations** of predicted Q (left) and V (right) values with their corresponding Monte Carlo return estimates.

6 Conclusion

We presented SAPO, a single-rollout autoregressive PPO framework that unifies policy, state-value, and action-value learning in one causal language model. By reading V and Q at the pre- and post-action boundaries and jointly optimizing policy, value, and on-policy SARSA objectives, SAPO retains temporal credit assignment without a separate critic or group-based rollouts. Across ALFWorld and WebShop with Qwen2.5-1.5B/7B and Qwen3-14B, SAPO matches standard PPO, substantially outperforms GRPO, and remains stable across random seeds while reducing peak memory and per-iteration runtime. Ablations and value-return correlations validate the shared value objectives. Under the evaluated benchmarks, model scales, and training settings, SAPO thus provides a low-cost PPO training paradigm with no observable performance loss. These results suggest that actor–critic optimization does not require duplicated policy-scale models, making PPO-style post-training more practical for long-horizon LLM agents.

AI USE STATEMENT

We used generative AI tools to assist with language editing, paraphrasing, and improving the clarity and readability of the manuscript. All AI-assisted revisions were reviewed and edited by the authors to ensure that they accurately reflected the intended technical meaning and did not introduce unsupported claims. The authors take full responsibility for the final content of this work, including all text, claims, and artifacts produced with the assistance of generative AI.

REPRODUCIBILITY STATEMENT

To support reproducibility, we describe SAPO and its optimization procedure in the main text. The experimental setup and appendix document the base models, training configurations, reward design, and computational resources used in our experiments. The appendix also provides the prompt templates for ALFWorld and WebShop, specifying the information available to the agent and the required response format. Together, these descriptions provide the implementation and evaluation details needed to reproduce our results.

REFERENCES

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report, 2023.

Arash Ahmadian, Chris Cremer, Matthias Gallé, Marzieh Fadaee, Julia Kreutzer, Olivier Pietquin, Ahmet Üstün, and Sara Hooker. Back to basics: Revisiting REINFORCE-style optimization for learning from human feedback in LLMs. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12248–12267, Bangkok, Thailand, 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.662. URL https://aclanthology.org/2024.acl-long.662/.

Xin Cheng, Shuo He, Lang Feng, HaiYang Xu, Ming Yan, Lei Feng, and Bo An. Beyond trajectory-level attribution: Graph-based credit assignment for agentic reinforcement learning. *arXiv preprint arXiv:2605.26684*, 2026.

Yunho Choi, Jongwon Lim, Woojin Ahn, Minjae Oh, Jeonghoon Shim, and Yohan Jo. Your language model is its own critic: Reinforcement learning with value estimation from actor's internal states, 2026. URL https://arxiv.org/abs/2605.07579.

Lang Feng, Zhenghai Xue, Tingcong Liu, and Bo An. Group-in-group policy optimization for llm agent training, 2026.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, et al. DeepSeek-R1 incentivizes reasoning in LLMs through reinforcement learning. *Nature*, 645:633–638, 2025. doi: 10.1038/s41586-025-09422-z. URL https://doi.org/10.1038/s41586-025-09422-z.

Zhenyu Hou, Yujiang Li, Jie Tang, and Yuxiao Dong. Single-rollout asynchronous optimization for agentic reinforcement learning, 2026. URL https://arxiv.org/abs/2607.07508.

Jingcheng Hu, Yinmin Zhang, Qi Han, Daxin Jiang, Xiangyu Zhang, and Heung-Yeung Shum. Open-Reasoner-Zero: An open source approach to scaling up reinforcement learning on the base model. In *Advances in Neural Information Processing Systems*, volume 38, pp. 162239–162262. Curran Associates, Inc., 2025. doi: 10.52202/085713-5418. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/ed873d79e7c268c020c4b4db13a2812a-Abstract-Conference.html.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Arik, Dong Wang, Hamed Zamani, and Jiawei Han. Search-R1: Training LLMs to reason and leverage search engines with reinforcement learning, 2025. URL https://arxiv.org/abs/2503.09516.

Amirhossein Kazemnejad, Milad Aghajohari, Eva Portelance, Alessandro Sordoni, Siva Reddy, Aaron Courville, and Nicolas Le Roux. VinePPO: Unlocking RL potential for LLM reasoning through refined credit assignment, 2024. URL https://arxiv.org/abs/2410.01679.

Kuan Li, Zhongwang Zhang, Huifeng Yin, Liwen Zhang, Litu Ou, Jialong Wu, Wenbiao Yin, Baixuan Li, Zhengwei Tao, Xinyu Wang, et al. Websailor: Navigating super-human reasoning for web agent, 2025a.

Xuefeng Li, Haoyang Zou, and Pengfei Liu. ToRL: Scaling tool-integrated reinforcement learning, 2025b.

Ziniu Li, Tian Xu, Yushun Zhang, Zhihang Lin, Yang Yu, Ruoyu Sun, and Zhi-Quan Luo. ReMax: A simple, effective, and efficient reinforcement learning method for aligning large language models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 29128–29163. PMLR, 2024. URL https://proceedings.mlr.press/v235/li24cd.html.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F. Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, volume 35, pp. 27730–27744. Curran Associates, Inc., 2022. doi: 10.52202/068431-2011. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/b1efde53be364a73914f58805a001731-Abstract.html.

Jiangweizhi Peng, Yuanxin Liu, Ruida Zhou, Charles Fleming, Zhaoran Wang, Alfredo Garcia, and Mingyi Hong. HiPER: Hierarchical plan–execute reinforcement learning for multi-turn LLM agents. In *Forty-third International Conference on Machine Learning*, 2026.

Michael Santacroce, Yadong Lu, Han Yu, Yuanzhi Li, and Yelong Shen. Efficient RLHF: Reducing the memory usage of PPO, 2023. URL https://arxiv.org/abs/2309.00754.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017. URL https://arxiv.org/abs/1707.06347.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models, 2024. URL https://arxiv.org/abs/2402.03300.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. Hybridflow: A flexible and efficient RLHF framework. In *Proceedings of the Twentieth European Conference on Computer Systems*, pp. 1279–1297, 2025.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning, 2023.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. Alfworld: Aligning text and embodied environments for interactive learning, 2020.

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F. Christiano. Learning to summarize with human feedback. In *Advances in Neural Information Processing Systems*, volume 33, pp. 3008–3021. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/hash/1f89885d556929e98d3ef9b86448f951-Abstract.html.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, et al. Gemini: a family of highly capable multimodal models, 2023.

Guoqing Wang, Sunhao Dai, Guangze Ye, Zeyu Gan, Wei Yao, Yong Deng, Xiaofeng Wu, and Zhenzhe Ying. Information gain-based policy optimization: A simple and effective approach for multi-turn search agents. In *The Fourteenth International Conference on Learning Representations*, 2026a.

Jiawei Wang, Jiacai Liu, Yuqian Fu, Yingru Li, Xintao Wang, Yuan Lin, Yu Yue, Lin Zhang, Yang Wang, and WANG KE. Harnessing uncertainty: Entropy-modulated policy gradients for long-horizon LLM agents. In *Forty-third International Conference on Machine Learning*, 2026b.

Zihan Wang, Kangrui Wang, Qineng Wang, Pingyue Zhang, Linjie Li, Zhengyuan Yang, Xing Jin, Kefan Yu, Minh Nhat Nguyen, Licheng Liu, et al. RAGEN: Understanding self-evolution in LLM agents via multi-turn reinforcement learning, 2025.

Zhiheng Xi, Jixuan Huang, Chenyang Liao, Baodai Huang, Jiaqi Liu, Honglin Guo, Yajie Yang, Rui Zheng, Junjie Ye, Jiazheng Zhang, et al. AgentGym-RL: An open-source framework to train LLM agents for long-horizon decision making via multi-turn reinforcement learning. In *The Fourteenth International Conference on Learning Representations*, 2026.

Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. Webshop: Towards scalable real-world web interaction with grounded language agents, 2022a.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models, 2022b.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Wang Zhang, Hang Zhu, Jinhua Zhu, Jiaze Chen, Jiangjie Chen, Chengyi Wang, Hongli Yu, Yuxuan Song, Xiangpeng Wei, Hao Zhou, Jingjing Liu, Wei-Ying Ma, Ya-Qin Zhang, Lin Yan, Mu Qiao, Yonghui Wu, and Mingxuan Wang. DAPO: An open-source LLM reinforcement learning system at scale, 2025. URL https://arxiv.org/abs/2503.14476.

Yufeng Yuan, Yu Yue, Ruofei Zhu, Tiantian Fan, and Lin Yan. What's behind PPO's collapse in long-CoT? value optimization holds the secret, 2025. URL https://arxiv.org/abs/2503.01491.

Yu Yue, Yufeng Yuan, Qiying Yu, Xiaochen Zuo, Ruofei Zhu, Wenyuan Xu, Jiaze Chen, Chengyi Wang, Tiantian Fan, Zhengyin Du, Xiangpeng Wei, Xiangyu Yu, Gaohong Liu, Juncai Liu, Lingjun Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Chi Zhang, Mofan Zhang, Wang Zhang, Hang Zhu, Ru Zhang, Xin Liu, Mingxuan Wang, Yonghui Wu, and Lin Yan. VAPO: Efficient and reliable reinforcement learning for advanced reasoning tasks, 2025. URL https://arxiv.org/abs/2504.05118.

Kaichen Zhang, Yuzhong Hong, Junwei Bao, Hongfei Jiang, Yang Song, Hong Dingqian, and Hui Xiong. GVPO: Group variance policy optimization for large language model post-training, 2025. URL https://arxiv.org/abs/2504.19599.

Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui Chen, Bowen Yu, Chang Gao, Kai Dang, Yuqiong Liu, Rui Men, An Yang, et al. Group sequence policy optimization, 2025a.

Yuxiang Zheng, Dayuan Fu, Xiangkun Hu, Xiaojie Cai, Lyumanshan Ye, Pengrui Lu, and Pengfei Liu. Deepresearcher: Scaling deep research via reinforcement learning in real-world environments. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 414–431, 2025b.

Yifei Zhou, Andrea Zanette, Jiayi Pan, Sergey Levine, and Aviral Kumar. ARChER: Training language model agents via hierarchical multi-turn reinforcement learning, 2024.

A SAPO TRAINING ALGORITHM

Algorithm 1 presents the complete SAPO training procedure. The rollout parameters remain fixed while trajectories and learning targets are constructed. Policy and value predictions are then recomputed together by the current model and optimized through one shared update.

Algorithm 1 Single-Rollout Autoregressive Policy Optimization

Input trainable model f_θ, reference policy π_{ref}, task distribution $\mathcal{D}$, and optimization hyperparameters
for training iteration $k = 1, \ldots, K$ **do**
  Set $\theta_{\text{old}} \leftarrow \theta$ and sample a task batch $\mathcal{B} \sim \mathcal{D}$
  for each task $\boldsymbol{x} \in \mathcal{B}$ **do**
    Sample one trajectory $\boldsymbol{\tau}$ with $\pi_{\theta_{\text{old}}}$ over the action vocabulary $\mathcal{W}_{\text{act}}$
    for each turn $t = 1, \ldots, T$ in $\boldsymbol{\tau}$ **do**
      Store the old token log probabilities, reward r_t, and termination indicator d_t
      Read V_t^{old} before the action and Q_t^{old} after the action from the shared value basis
    end for
    Set $V_{T+1}^{\text{old}} = Q_{T+1}^{\text{old}} = A_{T+1}^{\text{GAE}} = 0$
    for $t = T, \ldots, 1$ **do**
      Compute TD error: $\delta_t \leftarrow r_t + \gamma(1 - d_t)V_{t+1}^{\text{old}} - V_t^{\text{old}}$
      Compute GAE: $A_t^{\text{GAE}} \leftarrow \delta_t + \gamma\lambda(1 - d_t)A_{t+1}^{\text{GAE}}$
      Compute V target: $y_t^V \leftarrow V_t^{\text{old}} + A_t^{\text{GAE}}$
      Compute Q target: $y_t^Q \leftarrow r_t + \gamma(1 - d_t)Q_{t+1}^{\text{old}}$
    end for
  end for
  Apply the invalid action adjustment when available and normalize advantages over all valid turns in $\mathcal{B}$
  Broadcast each normalized turn advantage $\widehat{A}_t$ to the valid tokens of its action
  Run one shared-model forward pass to obtain current policy log probabilities, V_θ, and Q_θ
  Compute $\mathcal{L}_{\text{clip}}$, $\mathcal{L}_V$, and $\mathcal{L}_Q$, scaling the value targets by $R_{\max}$
  Compute reference-policy regularization $\mathcal{L}_{\text{KL}}$ and policy entropy $\mathcal{H}$
  Update θ by minimizing $\mathcal{L}_{\text{clip}} + \beta\mathcal{L}_{\text{KL}} - c_H\mathcal{H} + c_V\mathcal{L}_V + c_Q\mathcal{L}_Q$
end for
Output optimized parameters θ

B IMPLEMENTATION DETAILS

B.1 TRAINING DETAILS AND HYPERPARAMETERS

We implement SAPO using verl-agent (GiGPO), a scalable framework built on VeRL for training LLM agents through multi-turn interactions. All experiments run on x86_64 systems with Ubuntu 20.04/22.04. Our compute infrastructure includes 6 NVIDIA H200 GPUs and two NVIDIA H20 clusters with 64 and 8 GPUs, respectively. We use Qwen2.5-1.5B-Instruct, Qwen2.5-7B-Instruct, and Qwen3-14B-Instruct as base models, with checkpoint sizes of approximately 3.1, 15.2, and 29.6 GB, respectively. We train Qwen3-14B-Instruct on 32 H20 GPUs, Qwen2.5-7B-Instruct on 2 H200 GPUs, and Qwen2.5-1.5B-Instruct on either 4 A40 GPUs or a single H200 GPU.

The specific settings for SAPO are as follows. The total training loss comprises the value loss and Q-loss, with weights of $c_V = 0.1$ and $c_Q = 0.1$, respectively, without further hyperparameter tuning. A temperature coefficient is used to adjust the sensitivity of the predicted value; a value of 1.0 was adopted for both ALFWorld and WebShop, again without further tuning. The `value_cliprange` for the predicted value $v(c)$ is set to 0.2, consistent with the standard `cliprange` used in PPO.

ALFWorld. We train on ALFWorld for 150 iterations with a batch size of 256. Each episode is capped at 50 interaction turns, with maximum response and prompt lengths of 512 and 2048 tokens, respectively. The rollout sampling temperature is set to 0.4 during training and 1.0 during validation. For group-based methods, each iteration samples 16 groups of 8 rollouts, using $16 \times 8 = 128$ parallel

Prompt Template for ALFWorld

You are an expert agent operating in the ALFRED embodied Environment. Your task is to: {task_description}. Prior to this step, you have already taken {step_count} step(s). Below are the most recent {history_length} observations and the corresponding actions you took: {action_history}. You are now at step {current_step} and your current observation is: {current_observation}. Your admissible actions of the current situation are: [{admissible_actions}].
Now it's your turn to take an action. You should first reason step-by-step about the current situation. This reasoning process MUST be enclosed within <think> </think> tags. Once you've finished your reasoning, you should choose an admissible action for current step and present it within <action> </action> tags.

Figure 4: The prompt template for ALFWorld environment.

Prompt Template for WebShop

You are an expert autonomous agent operating in the WebShop e-commerce environment. Your task is to: {task_description}. Prior to this step, you have already taken {step_count} step(s). Below are the most recent {history_length} observations and the corresponding actions you took: {action_history}. You are now at step {current_step} and your current observation is: {current_observation}. Your admissible actions for the current situation are: [{available_actions}]. Now it's your turn to take one action for the current step. You should first reason step-by-step about the current situation, then think carefully which admissible action best advances the shopping goal. This reasoning process MUST be enclosed within <think> </think> tags. Once you've finished your reasoning, you should choose an admissible action for current step and present it within <action> </action> tags.

Figure 5: The prompt template for WebShop environment.

environments. PPO uses the same number of parallel environments to ensure a comparable rollout scale. We set both lambda λ and gamma γ in the eneralized advantage estimation to 0.95 and the KL-divergence loss coefficient to 0.01. The maximum trajectory-level outcome reward is $R_{\max} = 10$, and each invalid action incurs a turn-level penalty of -0.1.

WebShop. We train on WebShop for 150 iterations with a batch size of 64 and a maximum of 15 interaction turns per episode. The response length is capped at 512 tokens, while the prompt length is capped at 4096 tokens to accommodate the longer webpage observations. We set both lambda λ and gamma γ in the generalized advantage estimation to 0.95, and we use the same KL-divergence loss coefficient (0.01), training and validation sampling temperatures, parallel rollout configuration, and reward design as in ALFWorld.

B.2 Environmental Prompts

Prompt Templates for ALFWorld and WebShop. Figure 4 shows the prompt template for ALFWorld. At each turn, the prompt includes the task description, the number of completed steps, a recent observation–action history, the current observation, and the admissible actions. This context helps the agent track changes in the environment and choose an action appropriate to the current state. The agent is instructed to enclose its step-by-step reasoning in `<think>...</think>` and return exactly one admissible action in `<action>...</action>`, separating the reasoning text from the command executed by the environment.

The WebShop template, shown in Figure 5, follows the same structure, providing the shopping objective, the current step index, a recent observation–action history, the current webpage observation, and the available actions. The history contextualizes previous searches, navigation steps, and product selections, while the available actions identify the valid operations at the current step. Using the same

output format, the agent explains its choice in `<think>...</think>` and places a single action that advances the shopping objective in `<action>...</action>`.

C Baseline Descriptions

We compare SAPO with prompting-based agents, standard policy-gradient algorithms, and recent methods designed for fine-grained credit assignment. The first group measures what can be achieved through inference-time reasoning alone, while the latter two evaluate SAPO against established and specialized RL training paradigms.

Prompting-based Agents. **GPT-4o** Achiam et al. (2023) and **Gemini-2.5-Pro** Team et al. (2023) are closed-source general-purpose models evaluated through prompting without task-specific RL training. They provide reference points for strong proprietary models operating without task-specific RL training. **ReAct** Yao et al. (2022b) interleaves natural-language reasoning with environment actions, allowing the model to update its plan after each observation. **Reflexion** Shinn et al. (2023) augments this interaction loop with verbal self-reflection: after an unsuccessful attempt, the model summarizes its errors and conditions subsequent attempts on this feedback. ReAct and Reflexion therefore improve behavior through prompting and test-time interaction rather than parameter updates.

Standard Policy-Gradient Baselines. **PPO** Schulman et al. (2017) is our principal actor–critic baseline. It trains a policy with the clipped surrogate objective and uses a separately learned value function with GAE to propagate delayed rewards across interaction turns. PPO provides a strong reference for explicit value learning and temporal credit assignment, but incurs the memory and computation required by an additional critic. **GRPO** Shao et al. (2024) removes the learned critic by sampling a group of trajectories for each task and standardizing their outcome rewards within the group. The resulting relative advantage is used to optimize the policy, reducing critic-related training cost at the expense of multiple rollouts and trajectory-level credit. **RLOO** Ahmadian et al. (2024) is another critic-free REINFORCE-style method. For each sampled trajectory, it uses the mean reward of the remaining trajectories as a leave-one-out baseline, reducing gradient variance without fitting a value model.

Fine-Grained Credit-Assignment Baselines. **GiGPO** Feng et al. (2026) combines an episode-level macro advantage with a step-level micro advantage obtained by grouping actions that originate from matched anchor states across trajectories. We report both its standard variant and the variant that omits standard-deviation normalization, denoted $\text{GiGPO}_{\text{w/ std}}$ and $\text{GiGPO}_{\text{w/o std}}$, respectively. **EMPG** Wang et al. (2026b) modulates policy gradients using step-wise predictive entropy and the final task outcome. It strengthens updates for confident correct actions, penalizes confident errors, and attenuates highly uncertain steps. **GraphGPO** Cheng et al. (2026) aggregates sampled trajectories into a state-transition graph and estimates each visited state's distance to a successful goal. These methods provide direct comparisons with approaches that derive denser step-level supervision from trajectory groups, predictive uncertainty, or graph structure, whereas SAPO obtains turn-level learning signals from a learned value function using a single rollout per task.

D AlfWorld Cases

ALFWorld Cases

(Turn1) User: \nYou are Qwen, created by Alibaba Cloud. You are a helpful assistant.\nuser\n\nYou are an expert agent operating in the ALFRED Embodied Environment.\nYour current observation is: -= Welcome to TextWorld, ALFRED! =-\n\nYou are in the middle of a room. Looking quickly around you, you see a cabinet 11, a cabinet 10, a cabinet 9, a cabinet 8, a cabinet 7, a cabinet 6, a cabinet 5, a cabinet 4, a cabinet 3, a cabinet 2, a cabinet 1, a coffeemachine 1, a countertop 2, a countertop 1, a diningtable 2, a diningtable 1, a drawer 3, a drawer 2, a drawer 1, a fridge 1, a garbagecan 1, a microwave 1, a sidetable 1, a sinkbasin 1, a stoveburner 4, a stoveburner 3, a stoveburner 2, a stoveburner 1, and a toaster 1.\n\nYour task is to: heat some apple and put it in sidetable.\nYour admissible actions of the current situation are: ['go to cabinet 1'\n 'go to cabinet 10'\n 'go to cabinet 11'\n 'go to cabinet 2'\n 'go to cabinet 3'\n 'go to cabinet 4'\n 'go to cabinet 5'\n 'go to cabinet 6'\n 'go to cabinet 7'\n 'go to cabinet 8'\n 'go to cabinet 9'\n 'go to coffeemachine 1'\n 'go to countertop 1'\n 'go to countertop 2'\n 'go to diningtable 1'\n 'go to diningtable 2'\n 'go to drawer 1'\n 'go to drawer 2'\n 'go to drawer 3'\n 'go to fridge 1'\n 'go to garbagecan 1'\n 'go to microwave 1'\n 'go to sidetable 1'\n 'go to sinkbasin 1'\n 'go to stoveburner 1'\n 'go to stoveburner 2'\n 'go to stoveburner 3'\n 'go to stoveburner 4'\n 'go to toaster 1'\n 'inventory'\n 'look'].\n\nNow it's your turn to take an action.\nYou should first reason step-by-step about the current situation. This reasoning process MUST be enclosed within <think> </think> tags. \nOnce you've finished your reasoning, you should choose an admissible action for current step and present it within <action> </action> tags.\n\nassistant\n

Assistant: <think>\nThe goal is to heat an apple and put it in the sidetable. I need to first find an apple, heat it, and then put it in the sidetable.\n</think>\n\n<action>go to diningtable 1</action>

(Turn2) User: You are Qwen, created by Alibaba Cloud. You are a helpful assistant.\nuser\n\nYou are an expert agent operating in the ALFRED Embodied Environment. Your task is to: heat some apple and put it in sidetable.\nPrior to this step, you have already taken 1 step(s). Below are the most recent 1 observations and the corresponding actions you took: [Observation 1: '-= Welcome to TextWorld, ALFRED! =-\n\nYou are in the middle of a room. Looking quickly around you, you see a cabinet 11, a cabinet 10, a cabinet 9, a cabinet 8, a cabinet 7, a cabinet 6, a cabinet 5, a cabinet 4, a cabinet 3, a cabinet 2, a cabinet 1, a coffeemachine 1, a countertop 2, a countertop 1, a diningtable 2, a diningtable 1, a drawer 3, a drawer 2, a drawer 1, a fridge 1, a garbagecan 1, a microwave 1, a sidetable 1, a sinkbasin 1, a stoveburner 4, a stoveburner 3, a stoveburner 2, a stoveburner 1, and a toaster 1.\n\nYour task is to: heat some apple and put it in sidetable.', Action 1: 'go to diningtable 1']\nYou are now at step 2 and your current observation is: You arrive at diningtable 1. On the diningtable 1, you see a bowl 1, a bread 1, a cup 1, a egg 1, a knife 1, a lettuce 2, and a mug 1.\nYour admissible actions of the current situation are: ['examine diningtable 1'\n 'go to cabinet 1'\n 'go to cabinet 10'\n 'go to cabinet 11'\n 'go to cabinet 2'\n 'go to cabinet 3'\n 'go to cabinet 4'\n 'go to cabinet 5'\n 'go to cabinet 6'\n 'go to cabinet 7'\n 'go to cabinet 8'\n 'go to cabinet 9'\n 'go to coffeemachine 1'\n 'go to countertop 1'\n 'go to countertop 2'\n 'go to diningtable 2'\n 'go to drawer 1'\n 'go to drawer 2'\n 'go to drawer 3'\n 'go to fridge 1'\n 'go to garbagecan 1'\n 'go to microwave 1'\n 'go to sidetable 1'\n 'go to sinkbasin 1'\n 'go to stoveburner 1'\n 'go to stoveburner 2'\n 'go to stoveburner 3'\n 'go to stoveburner 4'\n 'go to toaster 1'\n 'inventory'\n 'look'\n 'take bowl 1 from diningtable 1'\n 'take bread 1 from diningtable 1'\n 'take cup 1 from diningtable 1'\n 'take egg 1 from diningtable 1'\n 'take knife 1 from diningtable 1'\n 'take lettuce 2 from diningtable 1'\n 'take mug 1 from diningtable 1'].\n\nNow it's your turn to take an action.\nYou should first reason step-by-step about the current situation. This reasoning process MUST be enclosed within <think> </think> tags. \nOnce you've finished your reasoning, you should choose an admissible action for current step and present it within <action> </action> tags.

Assistant: <think>\nTo heat an apple, I first need to find an apple and heat

Figure 6: Case study of SAPO on ALFWorld using Qwen2.5-1.5B-Instruct.

ALFWorld Cases

it.\n</think>\n\n<action>go to countertop 1</action>

(Turn3) User: You are Qwen, created by Alibaba Cloud. You are a helpful assistant.\nuser\n\nYou are an expert agent operating in the ALFRED Embodied Environment. Your task is to: heat some apple and put it in sidetable.\nPrior to this step, you have already taken 2 step(s). Below are the most recent 2 observations and the corresponding actions you took: [Observation 1: '-= Welcome to TextWorld, ALFRED! =-\n\nYou are in the middle of a room. Looking quickly around you, you see a cabinet 11, a cabinet 10, a cabinet 9, a cabinet 8, a cabinet 7, a cabinet 6, a cabinet 5, a cabinet 4, a cabinet 3, a cabinet 2, a cabinet 1, a coffeemachine 1, a countertop 2, a countertop 1, a diningtable 2, a diningtable 1, a drawer 3, a drawer 2, a drawer 1, a fridge 1, a garbagecan 1, a microwave 1, a sidetable 1, a sinkbasin 1, a stoveburner 4, a stoveburner 3, a stoveburner 2, a stoveburner 1, and a toaster 1.\n\nYour task is to: heat some apple and put it in sidetable.', Action 1: 'go to diningtable 1']\n[Observation 2: 'You arrive at diningtable 1. On the diningtable 1, you see a bowl 1, a bread 1, a cup 1, a egg 1, a knife 1, a lettuce 2, and a mug 1.', Action 2: 'go to countertop 1']\nYou are now at step 3 and your current observation is: You arrive at countertop 1. On the countertop 1, you see a fork 1, a knife 3, a lettuce 3, a pan 2, and a potato 1.\nYour admissible actions of the current situation are: ['examine countertop 1'\n 'go to cabinet 1'\n 'go to cabinet 10'\n 'go to cabinet 11'\n 'go to cabinet 2'\n 'go to cabinet 3'\n 'go to cabinet 4'\n 'go to cabinet 5'\n 'go to cabinet 6'\n 'go to cabinet 7'\n 'go to cabinet 8'\n 'go to cabinet 9'\n 'go to coffeemachine 1'\n 'go to countertop 2'\n 'go to diningtable 1'\n 'go to diningtable 2'\n 'go to drawer 1'\n 'go to drawer 2'\n 'go to drawer 3'\n 'go to fridge 1'\n 'go to garbagecan 1'\n 'go to microwave 1'\n 'go to sidetable 1'\n 'go to sinkbasin 1'\n 'go to stoveburner 1'\n 'go to stoveburner 2'\n 'go to stoveburner 3'\n 'go to stoveburner 4'\n 'go to toaster 1'\n 'inventory'\n 'look'\n 'take fork 1 from countertop 1'\n 'take knife 3 from countertop 1'\n 'take lettuce 3 from countertop 1'\n 'take pan 2 from countertop 1'\n 'take potato 1 from countertop 1'].\n\nNow it's your turn to take an action.\nYou should first reason step-by-step about the current situation. This reasoning process MUST be enclosed within <think> </think> tags. \nOnce you've finished your reasoning, you should choose an admissible action for current step and present it within <action> </action> tags.

Assistant: <think>\nIn the previous steps, I have moved to the dining table 1 and observed the items present on it. There is a bowl, bread, cup, egg, knife, lettuce, and a mug. I did not encounter an apple yet and did not necessarily need to heat an apple in this step.\n</think>\n\n<action>go to diningtable 2</action>

(Turn4) User:You are Qwen, created by Alibaba Cloud. You are a helpful assistant.\nuser\n\nYou are an expert agent operating in the ALFRED Embodied Environment. Your task is to: heat some apple and put it in sidetable.\nPrior to this step, you have already taken 3 step(s). Below are the most recent 2 observations and the corresponding actions you took: [Observation 2: 'You arrive at diningtable 1. On the diningtable 1, you see a bowl 1, a bread 1, a cup 1, a egg 1, a knife 1, a lettuce 2, and a mug 1.', Action 2: 'go to countertop 1']\n[Observation 3: 'You arrive at countertop 1. On the countertop 1, you see a fork 1, a knife 3, a lettuce 3, a pan 2, and a potato 1.', Action 3: 'go to diningtable 2']\nYou are now at step 4 and your current observation is: You arrive at diningtable 2. On the diningtable 2, you see a apple 2, a butterknife 2, a butterknife 1, a plate 1, a soapbottle 2, a soapbottle 1, a spatula 1, a spoon 2, and a spoon 1.\nYour admissible actions of the current situation are: ['examine diningtable 2'\n 'go to cabinet 1'\n 'go to cabinet 10'\n 'go to cabinet 11'\n 'go to cabinet 2'\n 'go to cabinet 3'\n 'go to cabinet 4'\n 'go to cabinet 5'\n 'go to cabinet 6'\n 'go to cabinet 7'\n 'go to cabinet 8'\n 'go to cabinet 9'\n 'go to coffeemachine 1'\n 'go to countertop 1'\n 'go to countertop 2'\n 'go to diningtable 1'\n 'go to drawer 1'\n 'go to drawer 2'\n 'go to drawer 3'\n 'go to fridge 1'\n 'go to garbagecan 1'\n 'go to microwave 1'\n 'go to sidetable 1'\n 'go to sinkbasin 1'\n 'go to stoveburner 1'\n 'go to stoveburner 2'\n 'go to stoveburner 3'\n 'go to stoveburner 4'\n 'go to toaster 1'\n 'inventory'\n 'look'\n 'take apple 2 from diningtable 2'\n

Figure 7: Case study of SAPO on ALFWorld using Qwen2.5-1.5B-Instruct.

ALFWorld Cases

'take butterknife 1 from diningtable 2'\n 'take butterknife 2 from diningtable 2'\n 'take plate 1 from diningtable 2'\n 'take soapbottle 1 from diningtable 2'\n 'take soapbottle 2 from diningtable 2'\n 'take spatula 1 from diningtable 2'\n 'take spoon 1 from diningtable 2'\n 'take spoon 2 from diningtable 2'].\n\nNow it's your turn to take an action.\nYou should first reason step-by-step about the current situation. This reasoning process MUST be enclosed within <think> </think> tags. \nOnce you've finished your reasoning, you should choose an admissible action for current step and present it within <action> </action> tags.

Assistant: <think>\nThe current situation is that I am at diningtable 2, which has an apple 2, a butterknife 1, a butterknife 2, a plate 1, a soapbottle 1, a soapbottle 2, a spatula 1, and a spoon 1, 2. I have already observed the diningtable 2 and the available actions.\n</think>\n\n<action>take apple 2 from diningtable 2</action>

(Turn5) User: You are Qwen, created by Alibaba Cloud. You are a helpful assistant.\nuser\n\nYou are an expert agent operating in the ALFRED Embodied Environment. Your task is to: heat some apple and put it in sidetable.\nPrior to this step, you have already taken 4 step(s). Below are the most recent 2 observations and the corresponding actions you took: [Observation 3: 'You arrive at countertop 1. On the countertop 1, you see a fork 1, a knife 3, a lettuce 3, a pan 2, and a potato 1.', Action 3: 'go to diningtable 2']\n[Observation 4: 'You arrive at diningtable 2. On the diningtable 2, you see a apple 2, a butterknife 2, a butterknife 1, a plate 1, a soapbottle 2, a soapbottle 1, a spatula 1, a spoon 2, and a spoon 1.', Action 4: 'take apple 2 from diningtable 2']\nYou are now at step 5 and your current observation is: You pick up the apple 2 from the diningtable 2.\nYour admissible actions of the current situation are: ['examine apple 2'\n 'examine diningtable 2'\n 'go to cabinet 1'\n 'go to cabinet 10'\n 'go to cabinet 11'\n 'go to cabinet 2'\n 'go to cabinet 3'\n 'go to cabinet 4'\n 'go to cabinet 5'\n 'go to cabinet 6'\n 'go to cabinet 7'\n 'go to cabinet 8'\n 'go to cabinet 9'\n 'go to coffeemachine 1'\n 'go to countertop 1'\n 'go to countertop 2'\n 'go to diningtable 1'\n 'go to drawer 1'\n 'go to drawer 2'\n 'go to drawer 3'\n 'go to fridge 1'\n 'go to garbagecan 1'\n 'go to microwave 1'\n 'go to sidetable 1'\n 'go to sinkbasin 1'\n 'go to stoveburner 1'\n 'go to stoveburner 2'\n 'go to stoveburner 3'\n 'go to stoveburner 4'\n 'go to toaster 1'\n 'inventory'\n 'look'\n 'move apple 2 to diningtable 2'].\n\nNow it's your turn to take an action.\nYou should first reason step-by-step about the current situation. This reasoning process MUST be enclosed within <think> </think> tags. \nOnce you've finished your reasoning, you should choose an admissible action for current step and present it within <action> </action> tags

Assistant: <think>\nThe current situation is that I have picked up the apple 2 from the diningtable 2. I need to heat the apple and put it in sidetable 1.\n</think>\n\n<action>go to microwave 1</action>

(Turn6)... (Turn7)

(Turn8) User: You are Qwen, created by Alibaba Cloud. You are a helpful assistant.\nuser\n\nYou are an expert agent operating in the ALFRED Embodied Environment. Your task is to: heat some apple and put it in sidetable.\nPrior to this step, you have already taken 7 step(s). Below are the most recent 2 observations and the corresponding actions you took: [Observation 6: 'You arrive at microwave 1. The microwave 1 is closed.', Action 6: 'heat apple 2 with microwave 1']\n[Observation 7: 'You heat the apple 2 using the microwave 1.', Action 7: 'go to sidetable 1']\nYou are now at step 8 and your current observation is: You arrive at sidetable 1. On the sidetable 1, you see a apple 1, a dishsponge 1, a lettuce 1, a saltshaker 1, and a tomato 1.\nYour admissible actions of the current situation are: ['examine apple 2'\n 'examine sidetable 1'\n 'go to cabinet 1'\n 'go to cabinet 10'\n 'go to cabinet 11'\n 'go to cabinet 2'\n 'go to cabinet 3'\n 'go to cabinet 4'\n 'go to cabinet 5'\n 'go to cabinet 6'\n 'go to cabinet 7'\n 'go to cabinet 8'\n 'go to cabinet 9'\n 'go to coffeemachine 1'\n 'go to countertop 1'\n 'go to countertop 2'\n 'go to diningtable 1'\n 'go to diningtable 2'\n 'go to drawer 1'\n 'go to drawer 2'\n 'go to drawer 3'\n

Figure 8: Case study of SAPO on ALFWorld using Qwen2.5-1.5B-Instruct.

ALFWorld Cases

'go to fridge 1'\n 'go to garbagecan 1'\n 'go to microwave 1'\n 'go to sinkbasin 1'\n 'go to stoveburner 1'\n 'go to stoveburner 2'\n 'go to stoveburner 3'\n 'go to stoveburner 4'\n 'go to toaster 1'\n 'inventory'\n 'look'\n 'move apple 2 to sidetable 1'].\n\nNow it's your turn to take an action.\nYou should first reason step-by-step about the current situation. This reasoning process MUST be enclosed within <think> </think> tags. \nOnce you've finished your reasoning, you should choose an admissible action for current step and present it within <action> </action> tags.

Assistant: <think>\nThe current situation is that I am at sidetable 1, and I have already heat the apple 2 using the microwave 1. I need to put the heated apple 2 in sidetable 1.\n</think>\n\n<action>move apple 2 to sidetable 1</action>

$R(\tau) = 10$

Figure 9: Case study of SAPO on ALFWorld using Qwen2.5-1.5B-Instruct.