Graph-Enhanced Policy Optimization in LLM Agent Training

Anonymous ACL submission

Abstract

Multi-step LLM agents in interactive environments represent a crucial step toward long-horizon decision-making. To train such agents, group-based reinforcement learning is widely adopted, which reinforces trajectories with higher relative performance within the group. However, in most existing methods, every step within a trajectory and every trajectory with the same terminal reward receive identical credit, regardless of their actual contributions. Since different states play different structural roles in an online state-transition graph built from sampled trajectories, their impacts should be differentiated and converted into task-aware credit at both the step and trajectory levels. We therefore present Graph-Enhanced Policy Optimization (GEPO), a framework for dual-level structural credit assignment in multi-step LLM agent training. Specifically, GEPO derives a state-level Task-Conditioned Criticality score that combines topological betweenness on the state-transition graph with semantic similarity to the task prompt. Based on this score, trajectory-level credit is reshaped through a state-adaptive discount, while step-level credit is scaled by the criticality of its successor state. Experimental results show that GEPO outperforms the strongest baselines by 1.1% in success rate on ALFWorld, 3.2% on WebShop, and 3.8% on average across search-augmented QA tasks at the 7B scale. Compared with flat group-based methods, GEPO reduces across-seed variance and concentrates gradient signals on the most critical steps.

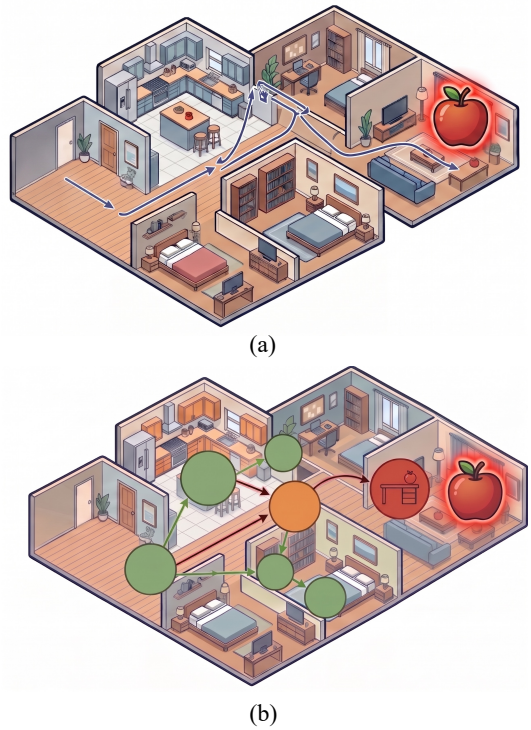

Figure 1: Comparison of credit assignment mechanisms. **(a)** Standard group-based methods suffer from **structural blindness**, uniformly distributing sparse rewards across all steps and causing gradient noise. **(b)** GEPO constructs an online topological graph to identify bottleneck states. This allows for redistributing the advantage precisely to the most critical step.

1 Introduction

Multi-step LLM agents (Yao et al., 2022b) in interactive environments represent a crucial step toward long-horizon decision-making. These agents (Chen et al., 2024) have been widely studied in tasks such as virtual home navigation (Shridhar et al., 2020), multi-step web shopping (Yao et al., 2022a; Furuta et al., 2023), and game-like world exploration (Wang et al., 2024a). The dominant approach for training such agents (Quadros et al., 2025) is group-based reinforcement learning (Shao et al., 2024), which compares trajectories sampled from the same prompt and reinforces those with higher relative performance. However, in most existing methods, every step within a trajectory and every trajectory with the same terminal reward receive identical credit, regardless of their actual contributions (Islam et al., 2022). In ALFWorld (Shridhar et al., 2020), for example, an agent must clear bottleneck states such as room transitions and con-

tainer transitions before any reward arrives, yet these pivotal steps receive no more credit than routine ones. This **structural blindness** inflates the variance of policy updates (Ying et al., 2024) and slows learning in topologically complex environments (Ramakrishnan et al., 2018).

The underlying topology of such environments can be recovered online by building a state-transition graph from each batch of sampled trajectories (Hazra and De Raedt, 2023). On this graph, individual steps and entire trajectories can be differentiated by the topological role of the states they visit, such as their betweenness centrality on the graph. This per-state structural signal can then be translated into task-aware credit at both the trajectory and step level.

Building on this idea, we introduce **Graph-Enhanced Policy Optimization (GEPO)**, a framework that performs dual-level structural credit assignment for LLM agent training. Each state receives a Task-Conditioned Criticality score that combines its betweenness centrality on the graph with its semantic similarity to the task prompt, so that states which are both structurally pivotal and task-relevant receive the highest scores. At the trajectory level, this score reshapes the return through a state-adaptive discount factor, allowing trajectories that reach the same reward via structurally different paths to receive distinguishable advantages. At the step level, the resulting trajectory advantage is redistributed onto individual actions and scaled by the successor state's criticality, so that steps leading to critical states are reinforced more strongly than those leading to routine ones.

Our contributions are as follows:

- To the best of our knowledge, we are among the first to introduce structural credit assignment through an online state-transition graph for LLM agent training, which adaptively reinforces critical steps and trajectories.

- We propose GEPO, a training framework that performs dual-level credit assignment at both the trajectory and step level using the state-transition graph. Each state on the graph receives a **Task-Conditioned Criticality** score that fuses betweenness centrality with semantic similarity to the task prompt. Based on this score, the trajectory's credit is reshaped through a state-adaptive discount, and each step's credit is scaled by its successor state's criticality.

- GEPO outperforms the strongest baselines by 1.1% in success rate on ALFWorld, 3.2% on WebShop (Yao et al., 2022a), and 3.8% on average across search-augmented QA tasks at the 7B scale. Compared with flat group-based methods, GEPO reduces across-seed variance and concentrates gradient signals on the most critical steps.

2 Related Work

Recent years have witnessed a rapid surge in efforts to develop LLM-based agents for complex multi-step tasks (Luo et al., 2025). Beyond single-step Q&A or text generation, these agents integrate large language models with environment interactions, enabling them to navigate virtual homes (Shridhar et al., 2020), browse the web for multiple steps (Yao et al., 2022a), and solve puzzle-like problems (Wang et al., 2024b). While such agents have demonstrated promising capabilities (Chen et al., 2024), they often face significant reward sparsity and partially observable states (Quadros et al., 2025), making reinforcement learning a natural choice for enabling long-horizon control (Erdogan et al., 2025).

A number of policy optimization methods have been adapted to LLMs, including PPO (Schulman et al., 2017) and DPO (Rafailov et al., 2023). However, these typically rely on dense rewards or train computationally expensive critic networks. Recent work has turned to group-based RL methods that learn from sparse feedback without a value network (Ouyang et al., 2022; Sharma et al., 2024; Lin et al., 2025). For instance, GRPO (Shao et al., 2024) compares trajectories within a group sharing the same prompt. While memory-efficient, GRPO distributes terminal rewards uniformly across all steps, exhibiting the **structural blindness** that injects high-variance gradient noise. To refine credit assignment, GiGPO (Feng et al., 2025) attempts empirical state matching across trajectories and uses the matched states directly as the unit of credit attribution. However, in combinatorially large textual environments, exact state overlaps are virtually nonexistent, and using approximate matches as the credit unit propagates matching errors directly into the step-level advantage. Similarly, recent extensions like HGPO (He et al., 2026) attempt to resolve historical context inconsistency through hierarchical history matching, which shares GiGPO's vulnerability to matching sparsity and does not recover the environment's topological bottlenecks.

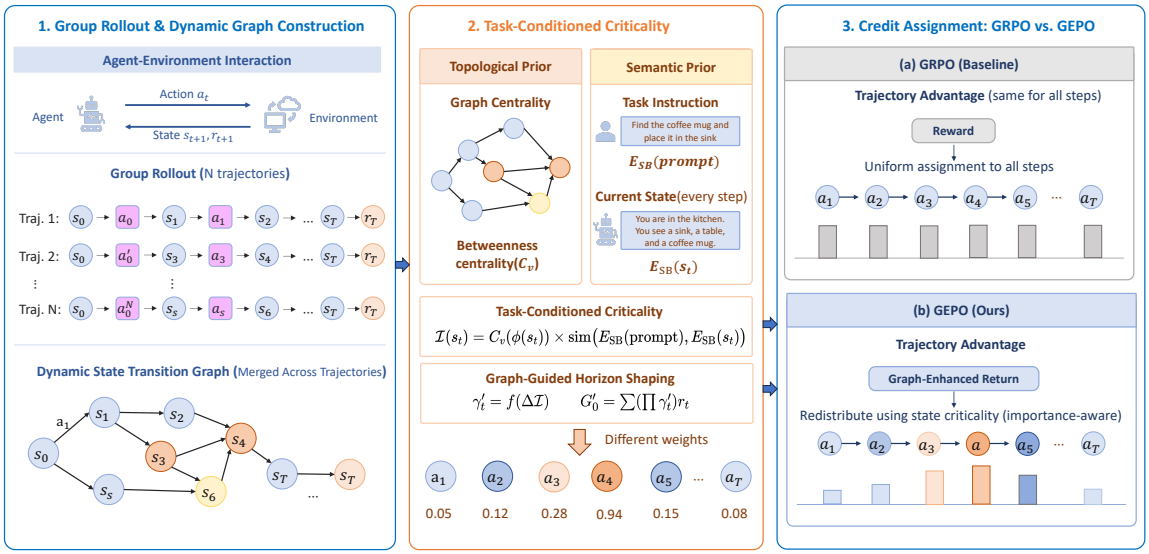

Figure 2: An overview of the GEPO framework. **(1)** The agent collects a group of trajectories through environment interaction, which are merged to construct a dynamic state-transition graph. **(2)** Task-Conditioned Criticality is computed by fusing betweenness centrality and semantic similarity, guiding the dynamic horizon shaping. **(3)** Unlike standard GRPO, which assigns uniform advantages across all steps, GEPO redistributes the trajectory-level advantage to individual steps based on their graph-derived criticality, enabling precise credit assignment.

GEPO also aggregates observations across trajectories, but only to construct a shared topological graph: credit is then derived from each state's structural role on that graph rather than from the matches themselves, so local matching noise affects credit only through second-order changes in graph centrality.

In parallel, graph-based RL methods have been used to improve exploration and credit assignment, e.g., by modeling observations as relational nodes or adding graph-based intrinsic rewards (Zambaldi et al., 2018; Hazra and De Raedt, 2023; Darvariu et al., 2024; Malysheva et al., 2020; McClusky, 2024; Xiong et al., 2025). However, they typically rely on static or predefined graphs and specialized GNNs, which are brittle to construct offline and ill-suited to the dynamically changing textual observations of LLM agents (Wang et al., 2023, 2025a).

In summary, group-based RL methods for LLM agents have so far either distributed rewards uniformly across steps or attempted cross-trajectory state matching in text spaces where such matches are unreliable, while graph-based RL methods depend on predefined graphs and specialized GNN architectures that are not well suited to dynamic textual observations. GEPO addresses both gaps with a single online state-transition graph that is built from the current batch of rollouts and used for dual-level credit assignment: it reshapes the trajectory-level return through a state-adaptive discount factor, and it redistributes the resulting trajectory advantage across steps according to the betweenness centrality of each successor state. The graph is constructed from textual observations directly, without intrinsic rewards or a separate GNN, and the credit assignment it induces is sign-preserving and variance-redistributing under standard PPO/GRPO advantage normalization, redistributing gradient mass from non-critical to bottleneck steps in proportion to the squared criticality factor.

3 Methodology

Figure 2 illustrates the GEPO framework. We first set up the POMDP (Kaelbling et al., 1998) and the group-based RL objective inherited from GRPO (Section 3.1), then construct an online state-transition graph from the sampled trajectories and score each state by a Task-Conditioned Criticality (Section 3.2). On top of this graph, GEPO performs dual-level credit assignment. At the trajectory level (Section 3.3), a state-adaptive discount factor conditioned on local criticality reshapes the return so that two trajectories reaching the same terminal reward through topologically distinct paths receive distinguishable advantage estimates. At the step level (Section 3.4), the resulting trajectory-level

advantage is redistributed onto individual actions and scaled by the criticality of each successor state.

3.1 Preliminaries

We consider a partially observable Markov decision process (POMDP) defined by the tuple $(\mathcal{S}, \mathcal{A}, p, r, \gamma)$, where an LLM agent generates textual actions $a_t \in \mathcal{A}$ based on textual observations $s_t \in \mathcal{S}$. Following the reasoning-acting paradigm (Yao et al., 2022b), an action a_t is not a primitive token but a complete textual sequence containing both the internal reasoning and the subsequent executable environment command. A trajectory is denoted as $\tau = (s_0, a_0, r_0, \ldots, s_T)$. Extrinsic rewards are typically sparse: r_t is zero for most steps and only becomes meaningful upon task completion, which makes long-horizon credit assignment difficult.

Group-based RL, such as GRPO (Shao et al., 2024), alleviates this challenge by learning from holistic trajectory comparisons without relying on computationally expensive value networks. Given a group of trajectories $\mathbb{G} = \{\tau_i\}_{i=1}^{|\mathbb{G}|}$ sharing the same initial prompt, GRPO optimizes a PPO-style surrogate objective. Denoting the importance sampling ratio as $r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}$, the objective is defined as:

$$\mathcal{L}_{\text{GRPO}} = \frac{1}{|\mathbb{G}|} \sum_{i=1}^{|\mathbb{G}|} \frac{1}{|\tau_i|} \sum_{t=1}^{|\tau_i|} \min\Big(r_t(\theta)\hat{A}_{i,t}, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_{i,t}\Big) \tag{1}$$

where ϵ is a clipping parameter, and $\hat{A}_{i,t}$ is the advantage estimate for the transition at step t of trajectory i.

This objective is memory-efficient, but obtaining a reliable step-level $\hat{A}_{i,t}$ remains the bottleneck. Standard GRPO in multi-step agent training uses a uniform trajectory-level advantage ($\hat{A}_{i,t} = \hat{A}_i$); this structural blindness produces high-variance gradient estimates and treats every step in a successful trajectory as equally responsible.

3.2 Dynamic Topological-Semantic Graph Construction

GEPO maintains an online state-transition graph as the topological prior over the environment. We call this construction the *Dynamic Topological-Semantic Graph* because it fuses dynamic topological updates from new rollouts with semantic state merging via Sentence-BERT (Reimers and Gurevych, 2019).

To handle state aliasing—where textually varied but semantically identical observations must collapse to the same vertex—we embed each observation with a pre-trained Sentence-BERT model targeting symmetric semantic search. An observation s_t is merged into the existing vertex with the maximum cosine similarity above a threshold δ; otherwise it forms a new vertex.

The graph is rebuilt every iteration, so $|V|$ does not grow across training steps. The S-BERT encoding and the topological updates computed via Brandes' algorithm (Brandes, 2001) incur moderate overhead compared with the LLM generation cost.

Once $\mathcal{G}$ is updated, we score each node by its task-conditioned criticality. Pure topological metrics can favor nodes that are central but task-irrelevant, so GEPO combines topology with semantic similarity to the task prompt. The Task-Conditioned Criticality $\mathcal{I}(s_t)$ is defined as:

$$\mathcal{I}(s_t) = C_v(\phi(s_t)) \times \text{sim}\big(E_{\text{SB}}(\text{prompt}), E_{\text{SB}}(s_t)\big) \tag{2}$$

where C_v is the topological betweenness centrality (Brandes, 2001) of the mapped node in $\mathcal{G}$, and $E_{\text{SB}}(\cdot)$ denotes the embedding extracted by the pre-trained Sentence-BERT model. The similarity term is clipped at 0 to ensure $\mathcal{I}(s_t) \geq 0$.

3.3 Trajectory-Level: Graph-Enhanced Advantage Formulation

We first describe how the graph reshapes the trajectory-level advantage. The standard return treats every transition along a trajectory uniformly through a fixed discount factor, so two trajectories that reach the same terminal reward are indistinguishable regardless of which states they visited. As the first level of credit assignment, we introduce a state-adaptive factor that ties reward propagation to local criticality on the graph; the resulting graph-enhanced return is then normalized in the standard group-wise way to obtain the trajectory-level advantage.

Graph-Guided Horizon Shaping. A fixed discount factor γ uniformly decays rewards across all transitions, failing to distinguish critical bottleneck states from incidental ones. To address this, we introduce a dynamic discount factor (François-Lavet et al., 2015) that adapts to how the criticality

changes from s_t to s_{t+1}:

$$\gamma'_t = \text{clip}\left(\gamma_{\text{base}} + w_\gamma \cdot \tanh(\Delta\mathcal{I}(t)), 0, 0.999\right), \tag{3}$$

where γ_{base} is a baseline discount, w_γ is a scaling factor, and $\Delta\mathcal{I}(t) = \mathcal{I}(s_{t+1}) - \mathcal{I}(s_t)$ is the change in criticality from s_t to s_{t+1}. Using $\Delta\mathcal{I}(t)$ rather than $\mathcal{I}(s_t)$ itself lets γ'_t react to *transitions* that enter or leave critical regions: $\gamma'_t > \gamma_{\text{base}}$ when the agent moves toward higher criticality and $\gamma'_t < \gamma_{\text{base}}$ when it moves away. The $\tanh$ keeps the adjustment bounded and smooth.

We then define the graph-enhanced return from time step t using the true extrinsic rewards r:

$$G'_t = \sum_{k=0}^{T-t} \left(\prod_{j=0}^{k-1} \gamma'_{t+j} \right) r_{t+k}. \tag{4}$$

In the common sparse-reward setting, Equation 4 simplifies to the terminal reward discounted by the cumulative factors $\prod \gamma'$. Consequently, two trajectories achieving identical success will receive distinct returns: an efficient path through critical bottleneck states experiences less decay than a redundant, meandering one. By shaping the horizon dynamically, GEPO differentiates trajectories by path quality without altering the true extrinsic objective or introducing biased intrinsic rewards.

Group-wise Normalization. Given a group of trajectories $\mathbb{G} = \{\tau_1, \ldots, \tau_N\}$ sampled from the same initial prompt, we compute their graph-enhanced returns $G'_0(\tau_i)$ and normalize within the group. With group mean $\mu_{\mathbb{G}}$ and standard deviation $\sigma_{\mathbb{G}}$, the trajectory-level advantage follows the standard mean-std form (Shao et al., 2024):

$$\hat{A}_{\text{traj}}(\tau_i) = \frac{G'_0(\tau_i) - \mu_{\mathbb{G}}}{\sigma_{\mathbb{G}} + \varepsilon}, \tag{5}$$

where $\varepsilon > 0$ ensures numerical stability. Because $G'_0(\tau_i)$ is shaped by the graph through γ'_t, two trajectories that reach the same terminal reward via topologically different paths can now receive different $\hat{A}_{\text{traj}}$. However, $\hat{A}_{\text{traj}}$ is still a single scalar per trajectory: applying it uniformly to every step gives redundant actions the same learning signal as pivotal ones, which motivates the step-level component in Section 3.4.

3.4 Step-Level: Structural Credit Redistribution

Although the trajectory-level advantage from Section 3.3 captures the graph-derived value of the path taken, it remains a single scalar per trajectory. As the second level of credit assignment, GEPO redistributes this advantage onto individual actions using the same online graph $\mathcal{G}$. Step-level credit is derived from each successor state's role in the current batch's transition structure, so it scales with how pivotal the action is within that batch rather than with a learned step-value.

We normalize the criticality scores of all states observed in the current group to a non-negative range $[0, 1]$, denoted $\tilde{\mathcal{I}}(\cdot)$. Following the standard convention that an action's value is tied to the state it reaches, we score action a_t by the criticality of its successor state s_{t+1}. The step-level advantage is then defined as a multiplicative redistribution of the trajectory-level advantage:

$$\hat{A}_t = \hat{A}_{\text{traj}}(\tau) \cdot \left(1 + \beta \cdot \tilde{\mathcal{I}}(s_{t+1})\right), \tag{6}$$

where $\beta \geq 0$ is a redistribution strength hyperparameter.

Because the scaling factor $(1 + \beta \cdot \tilde{\mathcal{I}}(s_{t+1}))$ is positive, $\hat{A}_t$ has the same sign as $\hat{A}_{\text{traj}}(\tau)$, so the policy gradient direction follows the trajectory outcome. Its magnitude is amplified for actions leading to high-criticality states and remains close to $\hat{A}_{\text{traj}}(\tau)$ for the rest.

Following standard practice in PPO/GRPO implementations, we further standardize the step-level advantages $\{\hat{A}_t\}_{(i,t)}$ to zero mean and unit variance across the batch. Together with the trajectory-level normalization (Equation 5), this batch-level standardization gives GEPO an exact variance-conservation property: the total per-step variance budget is preserved, while individual step variances are redistributed in proportion to $(1 + \beta\tilde{\mathcal{I}}(s_{t+1}))^2$, as formalized in Appendix B (Proposition 2).

The standardized $\hat{A}_t$ then replaces $\hat{A}_{i,t}$ in the surrogate objective (Equation 1) to form the GEPO objective $J_{\text{GEPO}}(\theta)$. The policy π_θ is updated by minimizing the composite loss:

$$\mathcal{L}(\theta) = -J_{\text{GEPO}}(\theta) + \beta_{\text{KL}} D_{\text{KL}}(\pi_\theta \| \pi_{\text{ref}}), \tag{7}$$

where β_{KL} is the penalty coefficient and D_{KL} is the Kullback-Leibler divergence (Schulman et al., 2017) between the active policy π_θ and a frozen reference model π_{ref}.

4 Experiments

We evaluate our method on three challenging benchmarks: ALFWorld, which tests long-horizon planning and spatial navigation; WebShop (Yao

Table 1: Performance comparison on the ALFWorld and WebShop benchmarks, averaged over 3 random seeds. For ALFWorld, we report success rates across its 6 subtasks and the overall result (%). For WebShop, we report both official score and success rate (%). GiGPO w/ std denotes using mean-std normalization, while GiGPO w/o std uses mean-centered without scaling. We follow the model scales established by prior work for each benchmark: 1.5B and 7B for ALFWorld and WebShop. Dashes indicate metrics not reported by the corresponding source; HGPO did not release subtask results.

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	25.4	58.7	60.3	42.5	35.9
Qwen2.5-1.5B-Instruct										
Prompting	Qwen2.5	5.9	5.5	3.3	9.7	4.2	0.0	4.1	23.1	5.2
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	PPO (with critic)	$64.8\pm_{3.5}$	$40.5\pm_{6.9}$	$57.1\pm_{4.9}$	$60.6\pm_{6.6}$	$46.4\pm_{4.0}$	$47.4\pm_{1.9}$	$54.4\pm_{3.1}$	$73.8\pm_{3.0}$	$51.5\pm_{2.9}$
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	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	GiGPO w/ std	$94.4\pm_{5.9}$	$67.5\pm_{3.6}$	$94.8\pm_{3.8}$	$\mathbf{94.4}\pm_{3.8}$	$79.8\pm_{4.7}$	$76.4\pm_{5.4}$	$86.7\pm_{1.7}$	$83.1\pm_{1.6}$	$65.0\pm_{3.2}$
RL Training	GiGPO 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	HGPO	-	-	-	-	-	-	$92.1\pm_{1.6}$	$90.6\pm_{1.1}$	$78.1\pm_{2.1}$
RL Training	**GEPO**	$\mathbf{99.2}\pm_{0.8}$	$\mathbf{84.5}\pm_{4.5}$	$\mathbf{96.8}\pm_{1.9}$	$88.5\pm_{5.2}$	$\mathbf{83.6}\pm_{5.1}$	$\mathbf{92.8}\pm_{3.8}$	$\mathbf{92.9}\pm_{1.1}$	$\mathbf{92.3}\pm_{1.4}$	$\mathbf{80.6}\pm_{2.0}$
Qwen2.5-7B-Instruct										
Prompting	Qwen2.5	33.4	21.6	19.3	6.9	2.8	3.2	14.8	26.4	7.8
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	PPO (with critic)	$92.3\pm_{4.0}$	$64.0\pm_{8.4}$	$92.5\pm_{2.4}$	$89.5\pm_{7.0}$	$80.3\pm_{2.0}$	$68.8\pm_{8.3}$	$80.4\pm_{2.7}$	$81.4\pm_{3.1}$	$68.7\pm_{5.1}$
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	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	GiGPO w/ std	$97.7\pm_{1.6}$	$82.7\pm_{7.9}$	$\mathbf{98.8}\pm_{1.6}$	$83.7\pm_{7.2}$	$89.3\pm_{5.2}$	$79.2\pm_{6.6}$	$90.8\pm_{1.3}$	$84.4\pm_{2.9}$	$72.8\pm_{3.2}$
RL Training	GiGPO w/o std	$91.8\pm_{5.4}$	$\mathbf{88.6}\pm_{6.3}$	$95.9\pm_{3.2}$	$90.2\pm_{2.6}$	$86.5\pm_{5.5}$	$85.2\pm_{4.7}$	$90.2\pm_{2.3}$	$86.2\pm_{2.6}$	$75.2\pm_{3.8}$
RL Training	HGPO	-	-	-	-	-	-	$95.2\pm_{1.0}$	$88.5\pm_{0.4}$	$79.3\pm_{1.2}$
RL Training	**GEPO**	$\mathbf{100.0}\pm_{0.0}$	$86.8\pm_{4.2}$	$98.6\pm_{2.4}$	$\mathbf{96.7}\pm_{2.0}$	$\mathbf{96.1}\pm_{2.9}$	$\mathbf{93.2}\pm_{2.6}$	$\mathbf{96.3}\pm_{1.2}$	$\mathbf{93.0}\pm_{2.0}$	$\mathbf{82.5}\pm_{1.2}$

et al., 2022a), which demands robust exploration within a vast, noisy HTML state space; and Search-Augmented QA (including NQ, HotpotQA, etc.) (Feng et al., 2025), which evaluates multi-step tool calling and complex information synthesis. We compare against prompting methods (ReAct, Reflexion) and RL baselines including PPO, RLOO, GRPO, GiGPO, and HGPO for ALFWorld and WebShop, and additionally R1-Instruct (DeepSeek-AI et al., 2025), Search-R1 (Jin et al., 2025), ZeroSearch (Sun et al., 2026), and StepSearch (Wang et al., 2025b) for search-augmented QA. We use the Qwen2.5 model family as the backbone across all experiments, with GPT-4o (OpenAI et al., 2023) and Gemini-2.5-Pro (Gemini Team Google: Rohan Anil et al., 2023) included as closed-source prompting references. Detailed descriptions, environment setups, and dataset details are provided in Appendix C.2.

4.1 Overall Performance

Table 1 and Table 2 compare GEPO with prompting baselines and recent RL methods (GRPO, GiGPO, HGPO) across three domains; all results are averaged over three random seeds. GEPO achieves the best aggregate score on every benchmark. On ALFWorld and WebShop it improves over the strongest baseline (HGPO) at both the 1.5B and 7B scales (Yang et al., 2024), and on search-augmented QA it improves over the strongest baseline by an average of $+3.8\%$ at the 7B scale. We discuss results along three axes matching the benchmarks.

Long-horizon planning under spatial bottlenecks (ALFWorld). ALFWorld requires the agent to traverse spatial bottlenecks before receiving any terminal reward, the regime where the structural blindness of flat group-based methods most amplifies gradient noise. Compared with GRPO and GiGPO, GEPO has both higher mean success rates and lower across-seed variance (± 1.2 overall at 7B), and reaches 100% on the 7B *Pick* subtask. The variance reduction is most pronounced against GRPO and GiGPO w/o std, and remains comparable to the more stable GiGPO w/ std variant. These results are consistent with the step-level redistribu-

Table 2: Performance on search-augmented QA tasks. GEPO is trained on NQ and HotpotQA, † and ⋆ indicate in-domain and out-of-domain datasets, respectively. We follow the model scales established by prior work for each benchmark: 3B and 7B for search-augmented QA.

Type	Method	Single-Hop QA			Multi-Hop QA				Avg.
		NQ†	TriviaQA⋆	PopQA⋆	HotpotQA†	2Wiki⋆	MuSiQue⋆	Bamboogle⋆	
Qwen2.5-3B-Instruct									
RL Training	R1-Instruct	27.0	53.7	19.9	23.7	29.2	7.2	29.3	27.1
RL Training	Search-R1	34.1	54.5	37.8	32.4	31.9	10.3	26.4	32.5
RL Training	ZeroSearch	41.4	57.4	44.8	27.4	30.0	9.8	11.1	31.7
RL Training	StepSearch	-	-	-	34.5	32.0	17.4	34.4	-
RL Training	GiGPO	42.0	59.5	42.4	36.9	**37.0**	12.6	64.1	42.1
RL Training	**GEPO**	**43.6**	**65.2**	**47.2**	**40.6**	36.6	**20.8**	**68.2**	**46.0**
Qwen2.5-7B-Instruct									
RL Training	R1-Instruct	21.0	44.9	17.1	20.8	27.5	6.0	19.2	22.4
RL Training	Search-R1	39.3	61.0	39.7	37.0	40.1	14.6	36.8	38.5
RL Training	ZeroSearch	43.6	61.8	51.5	34.6	35.2	18.4	27.8	39.1
RL Training	StepSearch	-	-	-	38.6	36.6	22.6	40.0	-
RL Training	GiGPO	46.4	64.7	46.1	41.6	43.6	18.9	68.9	47.2
RL Training	**GEPO**	**48.2**	**68.0**	**52.4**	**45.1**	**47.6**	**24.0**	**72.0**	**51.0**

tion: amplifying the advantage at successor states with high criticality reduces the contribution of non-critical steps to the gradient. GEPO achieves the best aggregate score on ALFWorld at both scales, although it does not dominate every individual subtask: at 1.5B it trails GiGPO w/ std on Heat (88.5 vs 94.4), and at 7B it trails GiGPO w/o std on Look by 1.8 points (86.8 vs 88.6) and GiGPO w/ std on Clean by 0.2 points (98.6 vs 98.8). We do not view scattered losses on individual subtasks as inconsistent with the overall claim: the aggregate score across all six subtasks is the primary metric, and GEPO leads it at both scales. For the 1.5B Heat case in particular, we observe that adding the state-adaptive discount and step-level redistribution on top of a small-capacity policy can introduce optimization noise that the model is not large enough to absorb; at 7B, where this effect disappears, GEPO leads on Heat by 6.5 points (96.7 vs 90.2).

Large textual state spaces (WebShop). WebShop contains a large HTML state space in which exact state overlaps across trajectories are rare. GEPO sidesteps this by deriving step-level credit from the online graph built within each batch, so it attains the highest success rates at both scales.

Multi-hop reasoning with long action chains (QA). In multi-hop QA the agent must chain multiple retrieval steps before the answer is rewarded, and a fixed discount factor attenuates the terminal signal before it reaches the early queries. The state-adaptive discount mitigates this attenuation, which is consistent with the gains on out-of-domain multi-hop sets such as Bamboogle ($+3.1\%$ over GiGPO at 7B).

Across all three settings, GEPO improves mean performance and reduces across-seed variance compared with flat group-based methods (GRPO, GiGPO), while also exceeding the strongest hierarchical baseline (HGPO) in mean performance with comparable variance.

4.2 Ablation Study

To evaluate the individual contributions of GEPO's mechanisms, we test three variants on the 7B model across ALFWorld, WebShop, and search-augmented QA tasks:

- **w/o Task Conditioning:** Uses raw topological betweenness C_v without semantic similarity.
- **w/o Credit Redistribution:** Reverts to a flat step-level advantage (disabling Equation 6).
- **w/o Horizon Shaping:** Uses a static discount factor ($\gamma = 0.90$) instead of the state-adaptive one (Equation 3).

As shown in Table 3, removing any component degrades performance, confirming their distinct roles:

Removing Credit Redistribution. This variant produces the largest drop on every benchmark, with -14.9% on ALFWorld, -14.2% on WebShop, and -9.8% on QA. Without redistribution, the

Table 3: Ablation study of GEPO components using the Qwen2.5-7B model. Results indicate the absolute success rate (%).

Model Variant	ALFWorld	WebShop	QA Avg.
Full GEPO	**96.3**	**82.5**	**51.0**
w/o Credit Redistribution	81.4 ± 1.2 (-14.9)	68.3 ± 1.5 (-14.2)	41.2 ± 1.4 (-9.8)
w/o Task Conditioning	89.5 ± 1.6 (-6.8)	73.2 ± 0.8 (-9.3)	43.8 ± 1.9 (-7.2)
w/o Horizon Shaping	92.1 ± 1.3 (-4.2)	76.5 ± 1.8 (-6.0)	45.2 ± 1.1 (-5.8)

trajectory-level advantage is broadcast uniformly to every step, so redundant actions receive the same learning signal as pivotal ones. The size of the drop confirms that the step-level redistribution is the main contributor to GEPO's gains over flat group-based methods.

Removing Task Conditioning. Replacing the Task-Conditioned Criticality with raw betweenness centrality drops performance most on WebShop (-9.3%), where instructions tightly constrain which states are relevant. The raw graph still identifies topologically central hubs, which are not necessarily task-relevant. The semantic similarity term keeps the criticality signal anchored to the current task prompt.

Removing Horizon Shaping. Replacing the state-adaptive discount with a fixed $\gamma = 0.90$ has the smallest effect of the three, with drops of -4.2% on ALFWorld, -6.0% on WebShop, and -5.8% on QA. The effect is largest on the two settings with longer action chains (WebShop and QA); the state-adaptive discount partially preserves the reward signal by increasing γ' as the agent moves toward more critical states.

4.3 Impact of Graph Scale on Performance

The group size n controls how many rollouts feed the online state-transition graph, directly affecting the reliability of the criticality score $\mathcal{I}(s_t)$. We vary n from 4 to 24 on WebShop with the 7B model. As shown in Table 4, success rate rises monotonically from 74.2% at $n = 4$ to 82.5% at $n = 16$. The graph grows sub-linearly in nodes ($\sim 3.4\times$ for a $4\times$ increase in n) while its edge-to-node ratio increases from 1.18 to 1.48, indicating that semantic merging consolidates redundant observations and that each vertex accumulates richer transition support, stabilizing the betweenness estimate.

Beyond $n = 16$, performance drops to 81.9% at $n = 20$ and 81.4% at $n = 24$. Two compounding effects drive the degradation: larger groups add topologically central but task-irrelevant states (e.g., generic product-listing pages in WebShop) that the semantic-similarity term only partially down-weights, and the $O(|V||E|)$ cost of betweenness on the denser graph grows faster than the marginal gain in coverage, pushing overhead to a prohibitive $+68\%$. We therefore use $n = 16$ as the default in all main experiments.

Table 4: Impact of the group size (n rollouts) on performance, average graph complexity, and computational cost. The experiment was conducted on the WebShop benchmark using the Qwen2.5-7B model. Graph complexity reflects the average number of nodes and edges in the online state-transition graph constructed per prompt during an update. Time Overhead indicates the additional wall-clock training time of GEPO compared to the GiGPO baseline with the same n. Optimal performance is in bold.

n	SR	Nodes	Edges	Overhead
4	74.2	36.2	42.8	+8%
8	78.5	68.5	89.3	+15%
12	81.1	97.1	135.6	+24%
16	**82.5**	**124.8**	**184.2**	**+35%**
20	81.9	149.3	233.7	+49%
24	81.4	171.6	281.5	+68%

5 Conclusion

We introduced Graph-Enhanced Policy Optimization (GEPO), a framework for training LLM agents in interactive environments. GEPO maintains an online state-transition graph built from the current batch of rollouts, and uses it for dual-level credit assignment: at the trajectory level, a state-adaptive discount factor reshapes each return according to local criticality on the graph; at the step level, the resulting trajectory-level advantage is redistributed onto individual actions scaled by the criticality of each successor state. Both levels share a single Task-Conditioned Criticality score that combines betweenness centrality with semantic similarity to the task. On ALFWorld, WebShop, and search-augmented QA, GEPO improves over the strongest baselines and reduces variance against flat group-based methods, while also exceeding the strongest hierarchical baseline (HGPO) in mean performance with comparable variance. These results suggest that graph-derived priors can effectively redirect gradient mass toward causally important steps when applied at both levels.

Limitations

GEPO has several limitations that we leave to future work.

Computational overhead. Building and analyzing the per-iteration graph adds wall-clock cost: on average $+35\%$ per training step relative to GiGPO, and up to $+68\%$ at large group sizes (Appendix D, Section 4.3). Because betweenness centrality scales as $O(|V||E|)$, scaling GEPO to very large rollout groups or environments with extremely long trajectories may require approximate centrality estimators.

Sensitivity to embedding quality. GEPO resolves state aliasing through pre-trained sentence embeddings, so its performance depends on how well the embedding model separates semantically distinct observations. In environments where important state changes correspond to small textual differences, the embedding may merge states that should be kept apart and produce a graph that misrepresents the environment's structure.

Single-agent scope. GEPO's credit assignment is defined for a single agent. Extending it to multi-agent settings would require deciding how the trajectory-level and step-level advantages are assigned across multiple interacting agents over long horizons, which we do not address here.

References

Arash Ahmadian, Chris Cremer, Matthias Gallé, Marzieh Fadaee, Julia Kreutzer, Ahmet Üstün, and Sara Hooker. 2024. Back to basics: Revisiting REINFORCE style optimization for learning from human feedback in LLMs.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, and 8 others. 2025. Qwen2.5-VL technical report.

Ulrik Brandes. 2001. A faster algorithm for betweenness centrality. `https://snap.stanford.edu/class/cs224w-readings/brandes01centrality.pdf`. Accessed: 2025-9-30.

Dingyang Chen, Qi Zhang, and Yinglun Zhu. 2024. Efficient sequential decision making with large language models.

Victor-Alexandru Darvariu, Stephen Hailes, and Mirco Musolesi. 2024. Graph reinforcement learning for combinatorial optimization: A survey and unifying perspective.

DeepSeek-AI, 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, and 181 others. 2025. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning.

Lutfi Eren Erdogan, Nicholas Lee, Sehoon Kim, Suhong Moon, Hiroki Furuta, Gopala Anumanchipalli, Kurt Keutzer, and Amir Gholami. 2025. Plan-and-Act: Improving planning of agents for long-horizon tasks.

Lang Feng, Zhenghai Xue, Tingcong Liu, and Bo An. 2025. Group-in-Group policy optimization for LLM agent training.

Vincent François-Lavet, Raphael Fonteneau, and Damien Ernst. 2015. How to discount deep reinforcement learning: Towards new dynamic strategies.

Hiroki Furuta, Kuang-Huei Lee, Ofir Nachum, Yutaka Matsuo, Aleksandra Faust, Shixiang Shane Gu, and Izzeddin Gur. 2023. Multimodal web navigation with instruction-finetuned foundation models.

Gemini Team Google: Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, Silver, and 1 others. 2023. Gemini: A family of highly capable multimodal models.

Rishi Hazra and Luc De Raedt. 2023. Deep explainable relational reinforcement learning: A neuro-symbolic approach.

Shuo He, Lang Feng, Qi Wei, Xin Cheng, Lei Feng, and Bo An. 2026. Hierarchy-of-Groups policy optimization for long-horizon agentic tasks.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. 2020. Constructing a multi-hop QA dataset for comprehensive evaluation of reasoning steps. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6609–6625, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Riashat Islam, Hongyu Zang, Manan Tomar, Aniket Didolkar, Md Mofijul Islam, Samin Yeasar Arnob, Tariq Iqbal, Xin Li, Anirudh Goyal, Nicolas Heess, and Alex Lamb. 2022. Representation learning in deep RL via discrete information bottleneck.

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

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, Vancouver, Canada. Association for Computational Linguistics.

Leslie Pack Kaelbling, Michael L. Littman, and Anthony R. Cassandra. 1998. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101(1):99–134.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466.

Mingxin Li, Yanzhao Zhang, Dingkun Long, Chen Keqin, Sibo Song, Shuai Bai, Zhibo Yang, Pengjun Xie, An Yang, Dayiheng Liu, Jingren Zhou, and Junyang Lin. 2026. Qwen3-vl-embedding and qwen3-vl-reranker: A unified framework for state-of-the-art multimodal retrieval and ranking. *arXiv preprint arXiv:2601.04720*.

Yufei Lin, Chengwei Ye, Huanzhen Zhang, Kangsheng Wang, Linuo Xu, Shuyan Liu, and Zeyu Zhang. 2025. CCL: Collaborative curriculum learning for sparse-reward multi-agent reinforcement learning via co-evolutionary task evolution.

Junyu Luo, Weizhi Zhang, Ye Yuan, Yusheng Zhao, Junwei Yang, Yiyang Gu, Bohan Wu, Binqi Chen, Ziyue Qiao, Qingqing Long, Rongcheng Tu, Xiao Luo, Wei Ju, Zhiping Xiao, Yifan Wang, Meng Xiao, Chenwu Liu, Jingyang Yuan, Shichang Zhang, and 7 others. 2025. Large language model agent: A survey on methodology, applications and challenges.

Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9802–9822, Toronto, Canada. Association for Computational Linguistics.

Aleksandra Malysheva, Daniel Kudenko, and Aleksei Shpilman. 2020. MAGNet: Multi-agent graph network for deep multi-agent reinforcement learning.

Ben McClusky. 2024. Dynamic graph communication for decentralised multi-agent reinforcement learning.

Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A Smith, and Mike Lewis. 2022. Measuring and narrowing the compositionality gap in language models.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, and 261 others. 2023. GPT-4 technical report.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L 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 Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback.

André Quadros, Cassio Silva, and Ronnie Alves. 2025. LLM-driven intrinsic motivation for sparse reward reinforcement learning.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D Manning, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model.

Ramya Ramakrishnan, Ece Kamar, Debadeepta Dey, Julie Shah, and Eric Horvitz. 2018. Discovering blind spots in reinforcement learning.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using siamese BERT-networks.

Max-Philipp B. Schrader. 2018. gym-sokoban. `https://github.com/mpSchrader/gym-sokoban`.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, Y K Li, Y Wu, and Daya Guo. 2024. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models.

Archit Sharma, Sedrick Keh, Eric Mitchell, Chelsea Finn, Kushal Arora, and Thomas Kollar. 2024. A critical evaluation of AI feedback for aligning large language models.

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

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

Hao Sun, Zile Qiao, Jiayan Guo, Xuanbo Fan, Yingyan Hou, Yong Jiang, Pengjun Xie, Yan Zhang, Fei Huang, and Jingren Zhou. 2026. ZeroSearch: Incentivize the search capability of LLMs without searching.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2021. MuSiQue: Multihop questions via single-hop question composition.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. 2024a. Voyager: An open-ended embodied agent with large language models. *Transactions on Machine Learning Research*.

Junyang Wang, Haiyang Xu, Haitao Jia, Xi Zhang, Ming Yan, Weizhou Shen, Ji Zhang, Fei Huang, and Jitao Sang. 2024b. Mobile-agent-v2: Mobile device operation assistant with effective navigation via multi-agent collaboration.

Mingyang Wang, Zhenshan Bing, Xiangtong Yao, Shuai Wang, Hang Su, Chenguang Yang, Kai Huang, and Alois Knoll. 2023. Meta-reinforcement learning based on Self-Supervised task representation learning.

Zihan Wang, Kangrui Wang, Qineng Wang, Pingyue Zhang, Linjie Li, Zhengyuan Yang, Xing Jin, Kefan Yu, Minh Nhat Nguyen, Licheng Liu, Eli Gottlieb, Yiping Lu, Kyunghyun Cho, Jiajun Wu, Li Fei-Fei, Lijuan Wang, Yejin Choi, and Manling Li. 2025a. RAGEN: Understanding self-evolution in LLM agents via multi-turn reinforcement learning.

Ziliang Wang, Xuhui Zheng, Kang An, Cijun Ouyang, Jialu Cai, Yuhang Wang, and Yichao Wu. 2025b. StepSearch: Igniting LLMs search ability via stepwise proximal policy optimization.

Siheng Xiong, Ali Payani, Yuan Yang, and Faramarz Fekri. 2025. Deliberate reasoning in language models as structure-aware planning with an accurate world model.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025. Qwen3 technical report.

Qwen: An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, and 23 others. 2024. Qwen2.5 technical report.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

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

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

Wangyang Ying, Haoyue Bai, Kunpeng Liu, and Yanjie Fu. 2024. Topology-aware reinforcement feature space reconstruction for graph data.

Vinicius Zambaldi, David Raposo, Adam Santoro, Victor Bapst, Yujia Li, Igor Babuschkin, Karl Tuyls, David Reichert, Timothy Lillicrap, Edward Lockhart, Murray Shanahan, Victoria Langston, Razvan Pascanu, Matthew Botvinick, Oriol Vinyals, and Peter Battaglia. 2018. Relational deep reinforcement learning.

Yuexiang Zhai, Hao Bai, Zipeng Lin, Jiayi Pan, Shengbang Tong, Yifei Zhou, Alane Suhr, Saining Xie, Yann LeCun, Yi Ma, and Sergey Levine. 2024. Fine-tuning large vision-language models as decision-making agents via reinforcement learning.

Yang Zhao, Yangou Ouyang, Xiao Ding, Hepeng Wang, Bibo Cai, Kai Xiong, Jinglong Gao, Zhouhao Sun, Li Du, Bing Qin, and Ting Liu. 2026. Consolidation or adaptation? PRISM: Disentangling SFT and RL data via gradient concentration.

A Pseudo Code

Algorithm 1 GEPO: Graph-Enhanced Policy Optimization

Initialize policy LLM π_θ, frozen S-BERT model, and the reference model $\pi_{\text{ref}} \leftarrow \pi_\theta$.
for each training iteration **do**
  Sample a group of trajectories $\mathbb{G} = \{\tau_1, \ldots, \tau_N\}$ using current policy π_θ.
  Initialize an empty directed graph $\mathcal{G}$ for this iteration.
  Add vertices and edges from $\mathbb{G}$ to $\mathcal{G}$: merge each observation s_t into the existing vertex with maximum S-BERT cosine similarity above threshold δ; otherwise create a new vertex.
  Compute Task-Conditioned Criticality $\mathcal{I}(s_t)$ for all visited nodes via Eq. 2.
  for each trajectory $\tau_i \in \mathbb{G}$ **do**
    for each timestep $t = 0, \ldots, T-1$ **do**
      Compute dynamic discount factor γ'_t from $\Delta\mathcal{I}(t)$ via Eq. 3.
    end for
    Compute graph-enhanced return $G'_0(\tau_i)$ via Eq. 4.
  end for
  Compute group-normalized trajectory advantage $\hat{A}_{\text{traj}}(\tau_i)$ across $\mathbb{G}$ via Eq. 5.
  for each trajectory $\tau_i \in \mathbb{G}$ **do**
    for each timestep t **do**
      Compute step-level advantage $\hat{A}_t$ from successor-state criticality $\tilde{\mathcal{I}}(s_{t+1})$ via Eq. 6.
    end for
  end for
  Standardize $\{\hat{A}_t\}_{(i,t)}$ to zero mean and unit variance across the batch (standard PPO/-GRPO advantage normalization).
  Update policy π_θ by minimizing the loss $\mathcal{L}(\theta)$ in Eq. 7.
end for

The GEPO training loop (Algorithm 1) proceeds in three phases at each iteration. (i) Graph construction and node scoring: the agent samples a group of trajectories, builds a fresh state-transition graph $\mathcal{G}$ from this batch via semantic vertex merging, and computes the Task-Conditioned Criticality $\mathcal{I}(s_t)$ for all visited nodes. (ii) Trajectory-level advantage computation: per-step dynamic discounts γ'_t are derived from criticality changes, yielding the graph-enhanced return $G'_0(\tau_i)$ and, after group normalization, the trajectory-level advantage $\hat{A}_{\text{traj}}(\tau_i)$. (iii) Step-level redistribution and policy update: the trajectory advantage is multiplicatively redistributed via $\tilde{\mathcal{I}}(s_{t+1})$ to produce $\hat{A}_t$, which is then standardized across the batch and used to update π_θ.

B Theoretical Analysis

We provide a formal analysis showing how GEPO addresses the bias–variance trade-off between trajectory-level and step-level advantage estimators in large combinatorial text spaces. Let A^*_t denote the oracle advantage for action a_t at state s_t, and let $c_t := 1 + \beta\tilde{\mathcal{I}}(s_{t+1}) \in [1, 1+\beta]$ denote the criticality scaling factor.

B.1 The Dilemma of Existing Estimators

Group-based methods such as GRPO use a trajectory-level advantage $\hat{A}_{\text{traj}}$. With a mean-centered group baseline, this estimator is unbiased in the sense that its sign matches the trajectory-level outcome relative to the group baseline. However, because the terminal reward is spread uniformly across all steps, $\hat{A}_{\text{traj}}$ inherits the noise of all other actions in the trajectory, so its variance $V_T = \mathrm{Var}[\hat{A}_{\text{traj}}]$ is large and credit is diluted.

To reduce variance, methods such as GiGPO compute a local advantage $\hat{A}_{\text{step}}$ by matching states across trajectories. In large text-based environments exact state overlaps are rare, so forcing semantic matches across distinct causal paths introduces a residual bias term, which we denote b_S for brevity: $\mathbb{E}[\hat{A}_{\text{step}}] = A^*_t + b_S$. When $|b_S| > |A^*_t|$ and the two terms have opposite signs the gradient direction is reversed, which penalizes correct actions.

B.2 Sign Preservation of GEPO

GEPO obtains step-level resolution through graph-based projection. Recall the credit redistribution rule

$$\hat{A}^{\text{GEPO}}_t = \hat{A}_{\text{traj}} \cdot c_t, \tag{8}$$

where $c_t = 1 + \beta\tilde{\mathcal{I}}(s_{t+1})$ and $\tilde{\mathcal{I}}(s_{t+1}) \in [0, 1]$, $\beta \geq 0$. Conditional on the state s_{t+1}, c_t is deterministic, so its expectation satisfies

$$\mathbb{E}\big[\hat{A}^{\text{GEPO}}_t \,\big|\, s_{t+1}\big] = \mathbb{E}\big[\hat{A}_{\text{traj}} \,\big|\, s_{t+1}\big] \cdot c_t. \tag{9}$$

Since $c_t \geq 1 > 0$,

$$\operatorname{sgn}\big(\mathbb{E}[\hat{A}_t^{\text{GEPO}} \mid s_{t+1}]\big) \equiv \operatorname{sgn}\big(\mathbb{E}[\hat{A}_{\text{traj}} \mid s_{t+1}]\big). \tag{10}$$

That is, the scaling is strictly multiplicative: it modulates magnitude without flipping the sign, and avoids the additive bias b_S inherent in horizontal matching.

B.3 Variance Conservation under Mean-Std Normalization

We now analyze how GEPO's redistribution rule, combined with the batch-level mean-std normalization standard in PPO/GRPO implementations (Section 3.4), shapes the per-step variance of the policy gradient.

Proposition 1 (Variance Redistribution)**.** *Conditional on c_t and treating $\hat{A}_{traj}$ as having conditional variance V_T, GEPO's per-step variance satisfies*

$$\operatorname{Var}\big[\hat{A}_t^{GEPO} \mid c_t\big] = c_t^2 \cdot V_T. \tag{11}$$

In particular, for two steps with criticality scores $\tilde{\mathcal{I}}(s_{r+1}) = u$ and $\tilde{\mathcal{I}}(s_{t+1}) = v$,

$$\frac{\operatorname{Var}[\hat{A}_t^{GEPO} \mid c_t]}{\operatorname{Var}[\hat{A}_r^{GEPO} \mid c_r]} = \left(\frac{1+\beta v}{1+\beta u}\right)^2, \tag{12}$$

which equals $(1+\beta)^2$ for a bottleneck step ($v = 1$) relative to a non-critical step ($u = 0$).

Proof. Direct from $\operatorname{Var}[X \cdot c] = c^2 \operatorname{Var}[X]$ for any constant scalar c and random variable X. □

Proposition 1 shows that GEPO concentrates variance at high-criticality steps, but does not by itself reduce absolute variance at any step (since $c_t \geq 1$). The complete picture emerges only after the batch-level mean-std normalization is applied.

Proposition 2 (Variance Conservation under Standardization)**.** *Let $\check{A}_t^{GEPO} := (\hat{A}_t^{GEPO} - \bar{A})/S$ denote the GEPO advantage after batch-level mean-std normalization, where $\bar{A}$ and S are the empirical mean and standard deviation of $\{\hat{A}_t^{GEPO}\}_{(i,t)}$ over the batch. Similarly, let $\check{A}_t^{GRPO} := \hat{A}_{traj}/\sqrt{V_T}$ denote the GRPO baseline after the same standardization. Assume:*

(a) $\mathbb{E}[\hat{A}_{traj}] = 0$ and $\operatorname{Var}[\hat{A}_{traj}] = V_T$ (from GRPO trajectory-level normalization, Equation 5).

(b) In the large-batch limit, the empirical correlation between c_t and $\hat{A}_{traj}$ becomes negligible relative to their marginal variances, justifying a standard decoupling approximation $\mathbb{E}[f(\hat{A}_{traj})\, g(c_t)] \approx \mathbb{E}[f(\hat{A}_{traj})]\, \mathbb{E}[g(c_t)]$ for the asymptotic analysis below.

Then, in the large-batch limit:

(i) Both estimators are marginally standardized: $\operatorname{Var}[\check{A}_t^{GRPO}] = \operatorname{Var}[\check{A}_t^{GEPO}] = 1$.

(ii) Conditional on c_t, the GEPO per-step variance is

$$\operatorname{Var}\big[\check{A}_t^{GEPO} \,\big|\, c_t\big] \approx \frac{c_t^2}{\mathbb{E}[c^2]}. \tag{13}$$

(iii) The total per-step variance budget is preserved:

$$\mathbb{E}_{c_t}\big[\operatorname{Var}\big[\check{A}_t^{GEPO} \,\big|\, c_t\big]\big] = 1 = \operatorname{Var}[\check{A}_t^{GRPO}]. \tag{14}$$

In particular, at non-critical steps ($c_t = 1$), $\operatorname{Var}[\check{A}_t^{GEPO}] \approx 1/\mathbb{E}[c^2] < 1$, while at bottleneck steps ($c_t = 1+\beta$), $\operatorname{Var}[\check{A}_t^{GEPO}] \approx (1+\beta)^2/\mathbb{E}[c^2] > 1$, with $\mathbb{E}[c^2] \in (1, (1+\beta)^2]$ whenever any $\tilde{\mathcal{I}}(s_{t+1}) > 0$.

Remark on assumption (b). Within a single trajectory, $\hat{A}_{\text{traj}}$ is a constant scalar while c_t varies across steps; the residual covariance between them comes only from cross-trajectory effects (e.g., successful trajectories visiting critical states more often). This residual is empirically small relative to $V_T \cdot \mathbb{E}[c^2]$ in our experimental setting, but we note that the proposition should be read as a first-order analytical model rather than an exact identity. The marginal variance equality in (i) and (iii) holds exactly by construction of the standardization; the conditional approximation in (ii) is what relies on (b).

Proof. Under assumption (b), we adopt the decoupling approximation $\mathbb{E}[f(\hat{A}_{\text{traj}})\, g(c_t)] \approx \mathbb{E}[f(\hat{A}_{\text{traj}})]\, \mathbb{E}[g(c_t)]$ in the large-batch limit, applied with both f, g as identity and as squaring. By assumption (a),

$$\bar{A} \approx \mathbb{E}[\hat{A}_{\text{traj}} \cdot c_t] \approx \mathbb{E}[\hat{A}_{\text{traj}}] \cdot \mathbb{E}[c_t] = 0, \tag{15}$$

$$S^2 \approx \mathbb{E}\big[\hat{A}_{\text{traj}}^2\, c_t^2\big] \approx \mathbb{E}[\hat{A}_{\text{traj}}^2] \cdot \mathbb{E}[c_t^2] = V_T \cdot \mathbb{E}[c^2]. \tag{16}$$

Thus $\check{A}_t^{\text{GEPO}} \approx \hat{A}_{\text{traj}}\, c_t / \sqrt{V_T \mathbb{E}[c^2]}$, and

$$\mathrm{Var}\big[\check{A}_t^{\text{GEPO}} \,\big|\, c_t\big] \approx \frac{c_t^2 \cdot V_T}{V_T \cdot \mathbb{E}[c^2]} = \frac{c_t^2}{\mathbb{E}[c^2]}, \quad (17)$$

which gives (ii). Taking the marginal expectation,

$$\mathbb{E}_{c_t}\left[\frac{c_t^2}{\mathbb{E}[c^2]}\right] = \frac{\mathbb{E}[c_t^2]}{\mathbb{E}[c^2]} = 1, \quad (18)$$

which gives (iii). For GRPO, $\check{A}_t^{\text{GRPO}} = \hat{A}_{\text{traj}}/\sqrt{V_T}$ has unit variance trivially, completing (i). □

Proposition 2 establishes that GEPO's step-level redistribution combined with standard mean-std normalization *redistributes* per-step variance rather than uniformly inflating or contracting it: the marginal variance budget per step matches that of vanilla GRPO by construction of the standardization, while the conditional variance is reallocated from non-critical to bottleneck steps in proportion to c_t^2. Together with the sign-preserving property of Section B.2, this yields a gradient estimator in which the direction matches the trajectory outcome, high-criticality steps receive a strictly larger share of the gradient magnitude than under flat group-based methods, and the additive bias b_S that destabilizes horizontal-matching estimators is replaced by a much milder second-order perturbation via the graph centrality.

C Experiment Details

Our experiments were conducted on two nodes with 16 NVIDIA L20 GPUs using the Qwen2.5 models (1.5B and 7B variants). We evaluated performance on the ALFWorld, WebShop, and search-augmented QA benchmarks, reporting success rates and official scores averaged over 3 random seeds. Table 5 summarizes the key hyperparameters used throughout the paper; further details on the experimental setup and datasets follow.

C.1 Experimental Settings

GEPO is benchmarked against two classes of methods: (i) prompting-based techniques, including ReAct (Yao et al., 2022b) and Reflexion (Shinn et al., 2023); and (ii) reinforcement learning algorithms, including PPO (Schulman et al., 2017), RLOO (Ahmadian et al., 2024), GRPO (Shao et al., 2024), GiGPO (Feng et al., 2025), and HGPO (He et al., 2026). All RL methods are trained with the same backbone (Qwen2.5-1.5B/7B for ALFWorld and WebShop; Qwen2.5-3B/7B for search-augmented QA), batch composition, and training budget, with method-specific hyperparameters tuned per the original papers.

C.2 Dataset Details

All datasets and models utilized in this study are publicly available and were used strictly for academic research, consistent with their intended purposes.

C.2.1 ALFWorld

The task in ALFWorld is to parse a high-level natural language goal into a sequence of low-level actions within a simulated household environment. The primary challenges are the long-horizon planning required to decompose the goal and the extreme reward sparsity, as meaningful feedback is provided almost exclusively upon successful completion of the entire multi-step task.

C.2.2 WebShop

The task in WebShop (Yao et al., 2022a) is to browse a realistic e-commerce website to find and purchase a product that satisfies a given user instruction. This requires the agent to perform a sequence of actions like searching for items, filtering attributes, and navigating through product pages. Key difficulties include a vast state space composed of noisy, high-dimensional HTML observations and a large action space of clickable elements, demanding robust exploration under sparse reward conditions.

C.2.3 Search-Augmented QA

To evaluate the multi-step tool calling and information synthesis capabilities of LLM agents, we follow the setup in GiGPO (Feng et al., 2025) and incorporate a variety of search-augmented Question Answering (QA) tasks. These benchmarks are categorized into two types: (1) **Single-hop QA**: including NQ (Kwiatkowski et al., 2019), TriviaQA (Joshi et al., 2017), and PopQA (Mallen et al., 2023), which require the agent to retrieve and identify a single piece of factual information. (2) **Multi-hop QA**: including HotpotQA (Yang et al., 2018), 2WikiMultiHopQA (Ho et al., 2020), MuSiQue (Trivedi et al., 2021), and Bamboogle (Ofir Press et al., 2022), which demand more complex reasoning as the agent must iteratively use search tools to link multiple pieces of evidence across different documents. The primary challenge in these tasks lies in the agent's ability to maintain a coherent reasoning chain and accurately

Table 5: Key hyperparameters in the GEPO framework.

Hyperparameter	Value	Description
Dynamic Graph Construction		
Semantic Similarity Threshold (δ)	0.9	Cosine similarity threshold for mapping textual observations to identical graph vertices.
Rollouts per Group (n)	16	The number of trajectories sampled per iteration, optimizing the bias-variance trade-off (see Section 4.3).
Graph Centrality Metric	Betweenness	The topological metric used to identify causal bottlenecks.
Graph-Guided Horizon Shaping		
Base Discount Factor (γ_{base})	0.90	The standard baseline discount factor for future rewards.
Dynamic Discount Weight (w_γ)	0.1	Scales the impact of the criticality difference ($\Delta\mathcal{I}$) on the discount factor (Equation 3).
Dynamic Discount Bounds	[0, 0.999]	Hard clipping range to ensure the adjusted discount factor remains mathematically stable.
Structural Credit Redistribution		
Redistribution Strength (β)	0.5	Controls the amplification magnitude of the step-level advantage based on normalized state criticality (Equation 6).
Advantage Normalization Epsilon (ε)	10^{-8}	Small constant added to the group standard deviation to prevent division by zero.
Policy Optimization (PPO-style)		
PPO Clipping Parameter (ϵ)	0.2	The standard clipping ratio to prevent destructively large policy updates.
KL Penalty Coefficient (β_{KL})	0.01	Weight of the KL divergence term in Equation 7 constraining the policy against π_{ref}.
Step-Level Advantage Normalization	mean-std	Standard PPO/GRPO advantage standardization applied to $\{\hat{A}_t\}$ across the batch before computing the surrogate loss.

formulate search queries to bridge information gaps over multiple turns.

C.3 Implementation Details and Hyperparameters

Table 5 lists the hyperparameters used throughout the paper. Below we add a few implementation notes that do not fit into the table.

To construct a topologically meaningful graph without succumbing to topology bloat, we use a pre-trained Sentence-BERT model. Observations are mapped to the same node if their cosine similarity strictly exceeds the threshold $\delta = 0.9$. This zero-shot semantic filtering is extremely lightweight and prevents minor textual variations from creating spurious nodes.

The transition graph is updated asynchronously as each trajectory rollout finishes, rather than batched at the end of an iteration. Because LLM generation has long-tail latency, the graph updates for early-completing trajectories overlap with the generation time of the slower ones, so by the time the rollout phase ends only the last trajectory is left to merge.

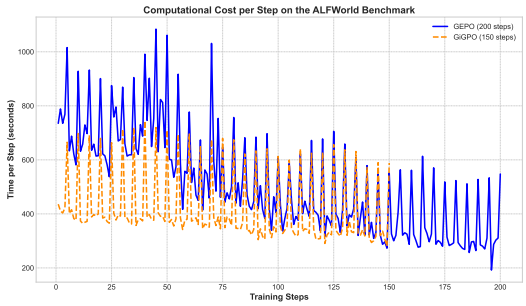

Figure 3: Comparison of computational cost per training step on the ALFWorld benchmark. The solid blue line (GEPO) is more expensive than the dashed orange line (GiGPO), with the gap concentrated in early training (first ∼75 steps) and shrinking as the graph stabilizes.

D Computational Cost Analysis

We compare the wall-clock training cost of GEPO and the GiGPO baseline in Figure 3. On average across the entire training process, GEPO incurs a computational overhead of 35% per step. However, this overhead is temporally asymmetric. During the first ∼75 steps, GEPO can be 50–80% slower. Once the graph stabilizes and a more competent policy produces shorter trajectories, the per-step cost converges, reducing the GEPO overhead to marginal levels in later stages. Overall, GEPO trades a moderate 35% training-time overhead for

the success-rate and stability gains reported in Section 4.1.

E Extension to Vision-Language Environments

Table 6: Success rates (%) of Vision-Language Model (VLM) agents using Qwen2.5-VL-3B-Instruct, averaged over 3 seeds. Best results are in **bold**.

Type	Method	Sokoban	EZPoints
RL	GRPO	67.1 $\pm$4.7	86.9 $\pm$3.4
RL	$\text{GiGPO}_{\text{w/ std}}$	76.9 $\pm$2.7	**100.0** $\pm$0.0
RL	$\text{GiGPO}_{\text{w/o std}}$	81.0 $\pm$3.6	**100.0** $\pm$0.0
RL	**GEPO (Ours)**	**85.0** $\pm$1.4	**100.0** $\pm$0.0

To verify GEPO's transferability to Vision-Language Models (VLMs), we evaluate it on Sokoban (Schrader, 2018) and EZPoints (Zhai et al., 2024) using Qwen2.5-VL-3B-Instruct (Bai et al., 2025). To process visual observations, we replace the Sentence-BERT text encoder with Qwen-3-VL-Embedding-8B (Li et al., 2026) and apply the same cosine-similarity merging rule to the resulting multimodal embeddings, leaving the rest of the GEPO pipeline unchanged.

As shown in Table 6, both GEPO and GiGPO saturate EZPoints at 100% success. On Sokoban, a puzzle environment with irreversible dead-ends, GEPO outperforms the stronger GiGPO variant (w/o std) by 4.0% with markedly lower variance. The gap is consistent with the analysis in Section 3.4: exact cross-trajectory state matches are rare in visual spaces, leaving horizontal-matching methods like GiGPO vulnerable to visual representation variations, while trajectory-level methods like GRPO fail to isolate early critical moves. By using cross-trajectory matches only to construct the graph topology rather than as the unit of credit attribution, GEPO confines the impact of visual matching noise to second-order changes in graph centrality, demonstrating its generality beyond text-only environments.

F Analysis of Topological Centrality Measures

We compare topological centrality metrics for the topological prior C_v on ALFWorld (Table 7). Betweenness yields the highest success rate and lowest variance at both model scales. This advantage arises because betweenness naturally isolates nodes lying on shortest paths, which in embodied environments closely align with causal bottlenecks. In contrast, degree centrality captures only local connectivity, eigenvector centrality emphasizes embeddedness in densely interconnected subnetworks, and closeness centrality minimizes average path length to all other nodes—none of which specifically identify the bottleneck transitions that betweenness highlights. Consequently, alternative metrics tend to spread credit across high-traffic but off-path nodes rather than concentrating it on causal bridges, leading to higher variance and sub-optimal credit assignment.

Table 7: Impact of Different Topological Centrality Measures on ALFWorld Performance across Model Scales. Overall success rates (%) are averaged over 3 seeds. Best results are in **bold**.

Centrality Metric	1.5B (%)	7B (%)
Betweenness (GEPO)	**92.9** $\pm$ **1.1**	**96.3** $\pm$ **1.2**
Eigenvector	89.5 $\pm$ 2.4	93.8 $\pm$ 2.1
Degree	87.2 $\pm$ 3.5	91.5 $\pm$ 4.0
Closeness	85.8 $\pm$ 3.8	89.4 $\pm$ 4.5

G Generality of GEPO on the Qwen3 Model Families

To evaluate generalization, we test GEPO on Qwen3 (1.7B and 8B) using the same search-augmented QA benchmarks as our main experiments. As shown in Table 8, GEPO consistently outperforms the strong GiGPO baseline, yielding average improvements of +2.8% (1.7B) and +3.1% (8B).

We highlight two key patterns: (1) Gains are most pronounced on tasks requiring longer reasoning chains (e.g., +4.2% on Bamboogle at 1.7B), validating GEPO's step-level redistribution over extended horizons. (2) The performance gap is consistent across both model sizes, suggesting that GEPO's graph-derived credit signal benefits search-augmented QA independently of the underlying model scale.

Finally, while Qwen3's absolute scores trail Qwen2.5 (aligning with concurrent findings on its agentic capabilities (Zhao et al., 2026)), GEPO's relative advantage over GiGPO remains robust across all evaluated settings.

Table 8: Performance comparison on search-augmented QA tasks using Qwen3-1.7B and Qwen3-8B (Yang et al., 2025). All metrics denote the evaluation success rate (%).

Model	Type	Method	Single-Hop QA			Multi-Hop QA				Avg.
			NQ	TriviaQA	PopQA	HotpotQA	2Wiki	MuSiQue	Bamboogle	
Qwen3-1.7B	RL Training	GiGPO	42.0	59.0	45.0	37.8	36.7	13.9	31.9	38.0
	RL Training	**GEPO (Ours)**	**44.1**	**62.2**	**47.4**	**40.3**	**38.8**	**16.5**	**36.1**	**40.8**
Qwen3-8B	RL Training	GiGPO	45.1	60.3	42.1	39.0	40.9	17.7	41.8	41.0
	RL Training	**GEPO (Ours)**	**47.9**	**63.8**	**45.2**	**41.6**	**43.7**	**20.9**	**45.3**	**44.1**

Table 9: A side-by-side comparison of agent trajectories in ALFWorld. Both agents are tasked with finding a clean soapbar and placing it in a cabinet. They share an identical history for the first two steps. The divergence at Step 3 highlights GEPO's superior efficiency and non-looping exploration strategy. Reasoning excerpts are verbatim from agent rollouts, abbreviated for space.

Step	GEPO (Task: put a clean soapbar in cabinet)	GiGPO (Task: put a clean soapbar in cabinet)
1	**Action:** go to countertop 1 **Observation:** Arrives at countertop 1, sees a candle 3 and a spraybottle 2.	**Action:** go to countertop 1 **Observation:** Arrives at countertop 1, sees a candle 3 and a spraybottle 2.
2	**Action:** go to toilet 1 **Observation:** Arrives at toilet 1, sees a toilet-paper 1.	**Action:** go to toilet 1 **Observation:** Arrives at toilet 1, sees a toilet-paper 1.
3	**Reasoning:** *"The task is to put a clean soapbar in the cabinet. I have already explored countertop 1 and toilet 1, and neither contains the target object. Based on the spatial structure, the next logical step is to check unvisited locations where a soapbar might be found. The action 'go to cabinet 1' is chosen as it is a new, promising location..."* **Action:** go to cabinet 1	**Reasoning:** *"The current situation is that you are at toilet 1, and there is no soapbar here. Your task is to put a clean soapbar in a cabinet. You have already taken the first step to go to countertop 1, but you might need to search that area more thoroughly to find the soapbar."* **Action:** go to countertop 1
	GEPO uses the explored portion of the graph to skip locations it has already checked, and moves to a new location (cabinet 1).	GiGPO does not retain this kind of graph-level memory and returns to countertop 1, a location it has already visited.

H Qualitative Case Study in ALFWorld

To qualitatively demonstrate how GEPO mitigates the structural blindness that affects standard group-based RL agents, we present a comparative case study from the ALFWorld environment. We analyze the trajectories of two agents—one trained with GEPO and a strong GiGPO baseline—assigned the same task and starting with the same initial exploration history. The task is to locate an item and place it in a cabinet. As shown in Table 9, both agents first explore countertop 1 and then toilet 1, finding neither location fruitful. The critical divergence occurs at Step 3.

Although the GiGPO agent acknowledges in its reasoning that countertop 1 has already been checked, it returns to that location at step 3. The agent does not carry a representation of the explored portion of the environment across reasoning steps, so the cue does not translate into a different action. The GEPO agent concludes that since the countertop and toilet have been inspected, it should try a new location and selects the cabinet. This is consistent with the criticality signal it was trained against: states that have already been visited contribute little to the redistributed advantage, while unvisited states with high task-conditioned criticality are favored.

I Prompts

Prompt Template for ALFWorld

You are an expert agent operating in the ALFWorld 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>** and **</think>** tags. Once you've finished your reasoning, you should choose an admissible action for current step and present it within **<action>** and **</action>** tags.

Figure 4: The prompt template of ALFWorld agents.

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>** and **</think>** tags. Once you've finished your reasoning, you should choose an admissible action for current step and present it within **<action>** and **</action>** tags.

Figure 5: The prompt template used for WebShop agents.

Prompt Template for Search

You are an expert agent tasked with answering the given question step-by-step. Your question: {task_description}. Prior to this step, you have already taken {step_count} step(s). Below is the interaction history where **<search>** **</search>** wrapped your past search queries and **<information>** **</information>** wrapped the corresponding search results returned by the external search engine. History: {memory_context}

Now it's your turn to respond for the current step. You should first conduct reasoning process. This process MUST be enclosed within **<think>** and **</think>** tags. After completing your reasoning, choose only one of the following actions (do not perform both):
(1) If you find you lack some knowledge, you can call a search engine to get more external information using format: **<search>** your query **</search>**.
(2) If you have enough knowledge to answer the question confidently, provide your final answer within **<answer>** **</answer>** tags, without detailed illustrations. For example, **<answer>**Beijing**</answer>**.

Figure 6: The prompt template of Search agents.