GRANULARITY AND CAPACITY MATCHING POLICY OPTIMIZATION FOR EFFICIENT AGENTIC RL

Anonymous authors
Paper under double-blind review

ABSTRACT

Training large language model (LLM) agents on long-horizon tasks requires identifying which intermediate decisions contribute to the final outcome. This is challenging when rewards are sparse and delayed. Existing methods trade off the granularity of credit assignment against the cost of value estimation. Token-level PPO provides fine-grained learning signals but incurs substantial critic training costs, whereas outcome-based GRPO assigns the same advantage to every token in a trajectory. We identify a granularity-capacity coupling, suggesting that value-estimation capacity should be matched to the temporal resolution of credit assignment. Based on this insight, we propose Granularity-Capacity Matched Proximal Policy Optimization (GCM-PPO), which jointly aligns credit-assignment granularity, critic capacity, and execution cost. GCM-PPO estimates values at agent-turn boundaries, providing distinct learning signals for successive decisions with substantially fewer value predictions. It employs an adaptive linear-plus-residual critic that reuses detached LLM representations, allowing lightweight CPU-side critic optimization to overlap with GPU-side actor optimization and rollout generation. By adapting critic capacity to actor scale and the available execution overlap, GCM-PPO reduces value-learning overhead while preserving turn-level credit assignment. Experiments across multiple benchmarks and model scales demonstrate that GCM-PPO matches or surpasses strong baselines while substantially improving training efficiency. Our anonymized code repository is available at https://anonymous.4open.science/r/GCM-PPO-C78D.

1 INTRODUCTION

Large language models (LLMs) (Brown et al., 2020; Achiam et al., 2023; Yang et al., 2025) are increasingly deployed as agents that solve tasks through multi-turn interactions with external environments (Yao et al., 2022b; Shinn et al., 2023; Schick et al., 2023). Unlike single-turn generation, these tasks involve sequences of decisions whose consequences may only become apparent after several subsequent interactions. While supervised fine-tuning (SFT) (Ouyang et al., 2022) learns from demonstration trajectories, reinforcement learning (RL) (Schulman et al., 2017) allows agents to improve through rewards obtained from environment feedback. In long-horizon tasks, however, rewards are often sparse and delayed, making it difficult to determine which intermediate decisions contributed to the final outcome (Andrychowicz et al., 2017; Arjona-Medina et al., 2019). A failed trajectory may contain useful actions, while a successful one may include ineffective steps. The central challenge is therefore credit assignment, which translates trajectory-level outcomes into learning signals for individual decisions (Zhang et al., 2025; Zhang, 2026). These signals are typically expressed as advantages, which indicate how much better or worse a decision is than expected.

Existing methods differ primarily in how they estimate advantages. Proximal Policy Optimization (PPO) (Schulman et al., 2017) typically trains a state-value model, known as a critic, to predict expected future rewards. Generalized advantage estimation (GAE) (Schulman et al., 2015) combines these predictions across successive states to compute token-level advantages, allowing different parts of a trajectory to receive different learning signals. However, this approach requires value predictions across numerous token prefixes and incurs substantial critic training costs. Group Relative Policy Optimization (GRPO) (Shao et al., 2024) avoids a learned critic by sampling multiple completions for each prompt and normalizing their rewards within the group. Under outcome-only supervision, every token in a completion receives the same advantage, making it difficult to distinguish

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.

the contributions of intermediate decisions. Subsequent methods provide more localized learning signals by comparing actions at shared states or constructing branching rollouts (Feng et al., 2026b; Ji et al., 2026). Other approaches improve value-estimation efficiency through turn-level modeling or lightweight critics (Liu et al., 2026a; Li et al., 2026a; Wang et al., 2026; Choi et al., 2026; Liang et al., 2026). However, existing methods largely treat credit-assignment granularity and critic capacity as separate design choices, leaving their relationship underexplored.

We investigate **granularity-capacity coupling**: how credit-assignment granularity affects the capacity needed for value estimation. Token-level estimation distinguishes fine-grained generation steps but requires numerous value predictions, while trajectory-level estimation avoids this overhead at the cost of assigning the same learning signal to all intermediate decisions. For interactive agents, turn-level estimation offers a natural middle ground, distinguishing successive agent decisions with substantially fewer value predictions (Li et al., 2026a). Whether this coarser granularity also permits a smaller critic remains an open question. Two observations motivate this possibility. **First**, the actor or reference LLM already encodes rich contextual information about the interaction history, allowing a compact critic to reuse these representations rather than learn them from scratch. **Second**, uncertainty in future policy decisions and environment responses may limit the marginal benefits of increased critic capacity, as effective variance reduction does not necessarily require highly accurate value estimates. These observations motivate a compact critic for value estimation.

Motivated by this coupling, we propose **G**ranularity-**C**apacity **M**atched Proximal Policy Optimization (**GCM-PPO**), a framework that jointly aligns credit-assignment granularity, critic capacity, and execution budget for efficient agentic RL. Specifically, GCM-PPO consists of three components: (i) **capacity-matched temporal abstraction**, which shifts value estimation from token prefixes to turn boundaries, providing distinct learning signals across agent turns with fewer value predictions, (ii) **adaptive residual value estimation**, which reuses detached LLM representations to train a compact linear-plus-residual critic whose capacity is adapted to actor scale, and (iii) **asynchronous lightweight critic optimization**, which exploits the compact architecture to perform CPU-side value learning in parallel with GPU-side actor optimization and rollout generation. GCM-PPO further adapts critic capacity to the available execution overlap window, reducing both the computational and scheduling overhead of value learning while retaining turn-level credit assignment. Our contributions are summarized as follows:

- We identify granularity-capacity coupling in agentic reinforcement learning, showing that capacity required for value estimation should match granularity of credit assignment.
- We propose GCM-PPO, which combines turn-level credit assignment, adaptive residual value estimation, and asynchronous lightweight critic optimization.
- Extensive experiments across multiple datasets and model scales show that GCM-PPO matches or surpasses strong baselines while substantially improving training efficiency.

2 PRELIMINARY

Episodic Policy Gradients. Given a parameterized policy π_θ, reinforcement learning maximizes the expected return over trajectories $\tau \sim p_\theta$ induced by the policy and environment, where $R(\tau) = \sum_t \gamma^t r_t$ is the discounted cumulative reward. The objective and its score-function gradient are

$$J(\theta) = \mathbb{E}_{\tau\sim p_\theta}[R(\tau)], \qquad \nabla_\theta J(\theta) = \mathbb{E}_{\tau\sim p_\theta}\left[R(\tau)\sum_t \nabla_\theta \log \pi_\theta(a_t \mid s_t)\right].$$

Assuming parameter-independent environment dynamics, this formulation estimates gradients from sampled trajectories without differentiating through state transitions. By causality, the full trajectory return can be replaced by the appropriately discounted reward-to-go for each action, excluding preceding rewards that the action cannot influence while retaining γ^t for the discounted objective.

Value Functions and Advantages. While reward-to-go removes irrelevant past rewards, policy-gradient estimates can still exhibit high variance over long horizons. The state-value and action-value functions characterize expected future returns under a given policy π:

$$V^\pi(s) = \mathbb{E}_\pi[G_t \mid s_t = s], \qquad Q^\pi(s,a) = \mathbb{E}_\pi[G_t \mid s_t = s, a_t = a],$$

where $G_t = \sum_{k=t}^{T-1} \gamma^{k-t} r_k$. Their difference, $A^\pi(s,a) = Q^\pi(s,a) - V^\pi(s)$, measures an action's expected return relative to the state value. Subtracting an action-independent, state-dependent baseline preserves the expected policy gradient because its expected score-weighted contribution is zero. This enables variance reduction through advantage estimation: PPO typically uses a learned value function and generalized advantage estimation, which combines exponentially weighted temporal-difference residuals to manage the bias–variance trade-off, whereas critic-free methods derive relative advantage estimates from sampled returns without training a separate value network.

3 METHOD

3.1 OVERVIEW

We propose GCM-PPO, a value-learning framework for efficient agentic reinforcement learning that matches critic capacity jointly to prediction granularity and the available execution budget. As shown in Figure 1, it consists of three components: (i) **Capacity-matched temporal abstraction** aligns credit assignment with the decision granularity of agent-environment interaction, reducing the resolution and capacity required for value prediction, (ii) **Adaptive residual value estimation** leverages this simpler prediction problem by reusing LLM representations and replacing the conventional model-scale value network with a compact residual critic whose capacity adapts to actor scale, and (iii) **Asynchronous lightweight critic optimization** exploits the resulting lightweight critic to decouple CPU-side value learning from GPU-side actor optimization and rollout generation, removing critic training from the critical execution path. This design preserves single-rollout advantage estimation while avoiding both model-scale critic optimization and grouped trajectory sampling. Further implementation details are provided in Appendix B.

3.2 CAPACITY-MATCHED TEMPORAL ABSTRACTION

The capacity required for value estimation depends directly on its temporal resolution. Token-level critics distinguish highly correlated autoregressive prefixes within each response, even though no new environment feedback is introduced between these intermediate states. We instead define value learning at turn boundaries, reducing the prediction resolution to the decision granularity of agent-environment interaction and correspondingly reducing the capacity required of the critic.

Turn-level interaction model. We formulate multi-turn agent interaction as a turn-level Markov decision process. At turn t, the state s_t contains the interaction history available to the agent, including previous model responses and environment observations. The agent generates an action $a_t = (y_{t,1}, \ldots, y_{t,L_t})$, consisting of the L_t tokens generated during the current turn. The environment then returns an observation o_t and an immediate reward r_t, yielding the next state s_{t+1}. We place the value function at these interaction boundaries and predict one value $V(s_t)$ for each turn.

Turn-level advantage estimation. Given predictions from the behavior critic $V_{\bar{\phi}}$, we compute generalized advantage estimates over turn-level transitions. The temporal difference residual is

$$\delta_t = r_t + \gamma(1 - d_t)V_{\bar{\phi}}(s_{t+1}) - V_{\bar{\phi}}(s_t),$$

where d_t indicates whether turn t terminates the trajectory. We then compute the advantage recursively as $\hat{A}_t = \delta_t + \gamma\lambda(1-d_t)\hat{A}_{t+1}$, and define the corresponding value target as $\hat{R}_t = \hat{A}_t + V_{\bar{\phi}}(s_t)$.

Turn-level policy optimization. We use the same temporal abstraction for policy optimization. Rather than applying the turn-level advantage to independently clipped token ratios, we aggregate token-wise log-probability changes into a length-normalized turn ratio,

$$\rho_t(\theta) = \exp\left(\frac{1}{L_t}\sum_{j=1}^{L_t}\left[\log \pi_\theta\left(y_{t,j} \mid s_t, y_{t,<j}\right) - \log \pi_{\theta_{\text{old}}}\left(y_{t,j} \mid s_t, y_{t,<j}\right)\right]\right).$$

This ratio is the geometric mean of token-wise likelihood ratios and measures the average per-token policy change over the complete turn. We apply PPO clipping once per turn,

$$\mathcal{L}_\pi(\theta) = -\frac{1}{|\mathcal{B}_{\text{turn}}|}\sum_{(\tau,t)\in\mathcal{B}_{\text{turn}}} \min\left(\rho_t(\theta)\hat{A}_t, \text{clip}\left(\rho_t(\theta), 1-\epsilon, 1+\epsilon\right)\hat{A}_t\right),$$

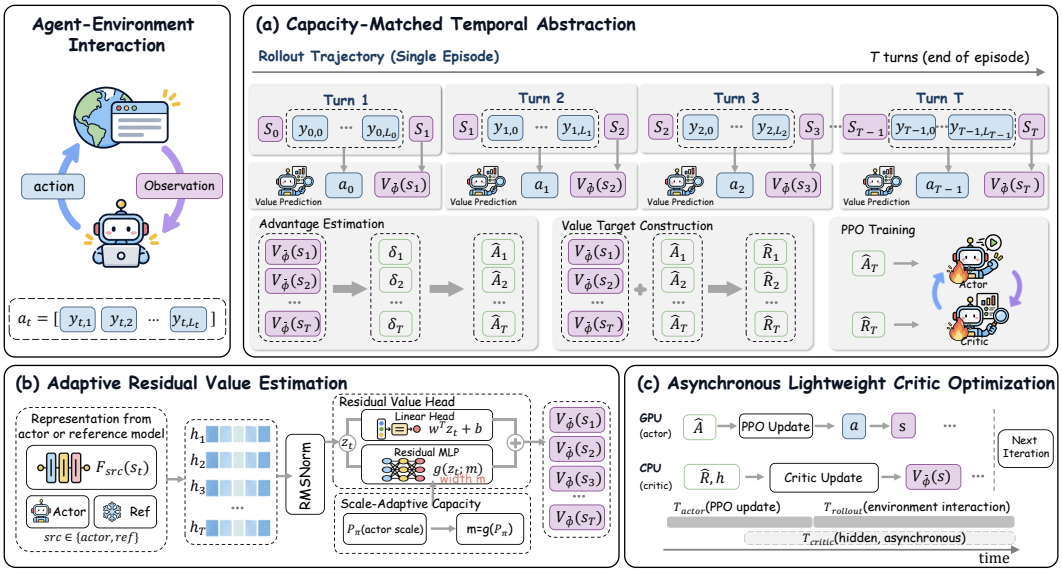

Figure 1: **Overview of GCM-PPO.** (i) **Capacity-Matched Temporal Abstraction:** Performs credit assignment at turn boundaries. (ii) **Adaptive Residual Value Estimation:** Estimates values from detached actor or reference representations with a compact linear-plus-residual critic. (iii) **Asynchronous Lightweight Critic Optimization:** Overlaps CPU-side critic updates with GPU-side actor optimization and rollout generation within a unified and efficient training pipeline.

where $\mathcal{B}_{\text{turn}}$ contains all valid turns in the training batch. Value prediction, advantage estimation, and policy clipping therefore operate at a common interaction-level temporal resolution. This consistency makes the value prediction problem sufficiently compact for a lightweight critic.

3.3 Adaptive Residual Value Estimation

Turn-level abstraction substantially reduces the resolution of value prediction, but a model-scale critic still duplicates the representation capacity of the LLM. We instead reuse interaction-boundary representations and restrict the trainable critic to a compact value-specific transformation.

For each interaction state s_t, let $h_t = F_{\text{src}}(s_t), \text{src} \in \{\text{actor}, \text{refer}\}$ denote the detached boundary representation produced by either the actor or the frozen reference model. Both variants share the same critic architecture and optimization objective, differing only in the source of h_t.

Residual value architecture. Because h_t already encodes the interaction state, the critic does not need to relearn a full state representation. We first normalize the reused representation as $\tilde{h}_t = \text{RMSNorm}(h_t)$, then decompose value prediction into a linear term and a nonlinear residual,

$$V_\phi(s_t) = \mathbf{w}^\top \tilde{h}_t + b + W_3\sigma\left(W_2\sigma\left(W_1\tilde{h}_t + b_1\right) + b_2\right) + b_3,$$

where σ denotes SiLU. The linear branch directly decodes value information represented in the LLM feature space, while the residual branch provides dedicated capacity for value-specific nonlinear structure. This parameterization avoids replicating the representation backbone and confines critic learning to the mapping from reusable LLM features to returns.

Scale-adaptive critic capacity. Let P_π denote the actor parameter count and m the hidden width of the residual branch. We instantiate the critic before RL training according to $m = g(P_\pi)$, where $g(\cdot)$ is a deterministic scale-dependent rule. The critic therefore varies its dedicated value capacity across actor scales while retaining the same residual parameterization. The admissible capacity at each scale is determined by the asynchronous execution constraint described in the next subsection.

Table 1: **Search QA performance across seven benchmarks.** GCM-PPO achieves the highest average performance on Qwen3-4B-Instruct-2507 and Qwen2.5-7B-Instruct. † and ∗ denote in-domain and out-of-domain datasets, respectively. Best results are bolded, second-best underlined.

Method	Single-Hop QA			Multi-Hop QA				Avg.
	NQ†	TriviaQA*	PopQA*	HotpotQA†	2Wiki*	MuSiQue*	Bamboogle*	
Qwen3-4B-Instruct-2507								
PPO	0.361	0.627	0.382	0.405	0.359	0.172	<u>0.452</u>	0.394
GRPO	0.395	<u>0.648</u>	0.419	0.416	0.396	0.187	0.435	0.414
GiGPO	0.380	0.641	0.412	0.386	0.397	0.168	0.435	0.403
Tree-GRPO	0.396	**0.653**	0.420	0.414	0.387	0.182	<u>0.452</u>	0.415
Turn-PPO	<u>0.409</u>	0.643	0.427	**0.423**	0.388	<u>0.193</u>	0.444	0.418
SPPO	0.407	0.629	0.426	**0.423**	**0.441**	**0.194**	<u>0.452</u>	<u>0.425</u>
POISE	0.357	0.589	0.390	0.395	0.365	0.173	**0.476**	0.392
Ours (actor)	**0.436**	0.636	**0.445**	<u>0.421</u>	0.409	**0.194**	0.435	<u>0.425</u>
Ours (refer)	**0.436**	0.647	<u>0.440</u>	0.405	<u>0.430</u>	0.177	**0.476**	**0.430**
Qwen2.5-7B-Instruct								
PPO	**0.445**	0.612	<u>0.440</u>	0.358	0.256	0.115	0.315	0.363
GRPO	0.422	0.617	0.409	<u>0.383</u>	0.301	0.163	0.379	0.382
GiGPO	0.404	<u>0.623</u>	0.418	0.357	<u>0.355</u>	0.153	0.371	0.383
Tree-GRPO	0.420	0.612	0.411	0.382	0.335	<u>0.168</u>	**0.403**	<u>0.390</u>
Turn-PPO	0.420	0.595	0.433	0.343	0.256	0.127	0.210	0.341
SPPO	0.424	0.580	0.422	0.370	0.312	0.136	0.363	0.372
POISE	0.352	0.553	0.362	0.277	0.251	0.082	0.250	0.304
Ours (actor)	<u>0.444</u>	**0.624**	<u>0.440</u>	**0.390**	**0.364**	**0.180**	<u>0.387</u>	**0.404**
Ours (refer)	0.419	**0.624**	**0.452**	0.349	0.254	0.117	0.347	0.366

Value optimization. We optimize the critic using the clipped PPO value objective. We define $V_\phi^{\text{clip}}(s_t) = V_{\bar{\phi}}(s_t) + \text{clip}\left(V_\phi(s_t) - V_{\bar{\phi}}(s_t), -\epsilon_v, \epsilon_v\right)$ and minimize

$$\mathcal{L}_V(\phi) = \frac{1}{2|\mathcal{B}_{\text{turn}}|} \sum_{(\tau,t)\in\mathcal{B}_{\text{turn}}} \max\left(\left(V_\phi(s_t) - \hat{R}_t\right)^2, \left(V_\phi^{\text{clip}}(s_t) - \hat{R}_t\right)^2\right),$$

where $V_{\bar{\phi}}(s_t)$ denote the value prediction recorded before the critic update.

3.4 Asynchronous Lightweight Critic Optimization

The compact residual critic can be optimized on CPU from detached LLM representations, allowing value learning to proceed independently of GPU-side policy computation. We exploit this property to remove critic optimization from the sequential execution path.

At iteration k, the current critic V_{ϕ_k} evaluates the collected trajectories and provides the predictions used to construct $\hat{A}_t$ and $\hat{R}_t$. The detached representations and value targets are transferred to CPU, where the critic is updated from ϕ_k to ϕ_{k+1}. In parallel, the GPU performs actor optimization on the current batch and subsequently generates the next rollout batch. The updated critic is required only when the next batch reaches value and advantage estimation.

This overlap directly determines the critic capacity available. We choose $g(P_\pi)$ such that

$$T_{\text{critic}}\left(g(P_\pi)\right) \lesssim T_{\text{actor}}(P_\pi) + T_{\text{rollout}}(P_\pi),$$

where T_{critic}, T_{actor}, and T_{rollout} denote critic optimization, actor optimization, and rollout generation time, respectively. Larger actor configurations provide a larger overlap window and can therefore accommodate wider residual critics without extending sequential iteration time.

When this constraint is satisfied, critic optimization is hidden behind GPU-side computation while actor optimization and rollout generation retain their original execution order.

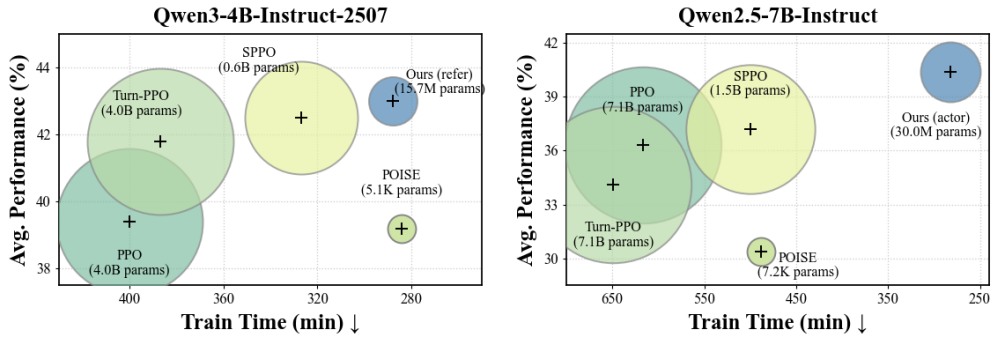

Figure 2: **Performance-efficiency comparison.** Bubble size represents critic parameter count, while the two axes report average QA performance and training time across model scales under comparable experimental settings. Better configurations lie toward the upper right with smaller bubbles, jointly indicating higher performance, shorter training time, and lower critic capacity.

The three components form a unified value learning design. Turn-level abstraction reduces the complexity of credit assignment to the natural interaction granularity. Representation reuse and residual modeling translate this simpler prediction problem into a compact critic. Asynchronous optimization then converts the reduced critic capacity into lower accelerator memory use and less sequential computation during RL training. The resulting framework preserves single-rollout advantage estimation while avoiding both a model-scale critic and grouped trajectory sampling.

4 Experiments

We conduct a comprehensive evaluation of **GCM-PPO** on search-augmented question answering across multiple model scales. Our experiments are designed to answer the following research questions. **RQ1 (Task Performance):** Does granularity–capacity matching outperform critic-based and critic-free policy optimization across task complexities and model scales? **RQ2 (Learning Efficiency):** How does granularity–capacity matching affect the performance–cost frontier, training dynamics, and robustness across representation sources? **RQ3 (Ablation Study):** Which components drive the gains from aligning temporal granularity, critic capacity, and policy optimization?

4.1 Experimental Setup

Datasets. We evaluate on seven open-domain question answering benchmarks. NQ, TriviaQA, and PopQA evaluate single-hop retrieval, while HotpotQA, 2WikiMultiHopQA, MuSiQue, and Bamboogle require multi-hop evidence aggregation. Following Search-R1, we train on NQ and HotpotQA and evaluate on all seven datasets, covering both in-domain performance and cross-dataset generalization under a unified evaluation protocol. Further details are provided in Appendix D.1.

Baselines. We compare with seven baselines spanning three types of methods: (i) **foundational policy optimization**, including PPO (Schulman et al., 2017) and GRPO (Shao et al., 2024), (ii) **fine-grained credit assignment**, including GiGPO (Feng et al., 2026b) and Tree-GRPO (Ji et al., 2026), and (iii) **efficient learned value estimation**, including Turn-PPO (Li et al., 2026a), SPPO (Wang et al., 2026), and POISE (Choi et al., 2026). Further details are provided in Appendix D.2.

Evaluation Metrics. We report normalized Exact Match (EM) accuracy on all datasets. Predictions and reference answers are normalized by lowercasing, removing punctuation and articles, and collapsing consecutive whitespace. A prediction is correct if it exactly matches any normalized reference answer. We use the same criterion to compute outcome rewards, aligning training supervision with final-answer evaluation. Further details are provided in Appendix D.3.

Implementation Details. We use `e5-base-v2` as the dense retriever over the December 2018 English Wikipedia corpus and retrieve the top three passages for each query. All experiments use the same retriever, corpus, and retrieval configuration to maintain a consistent evaluation setting. We

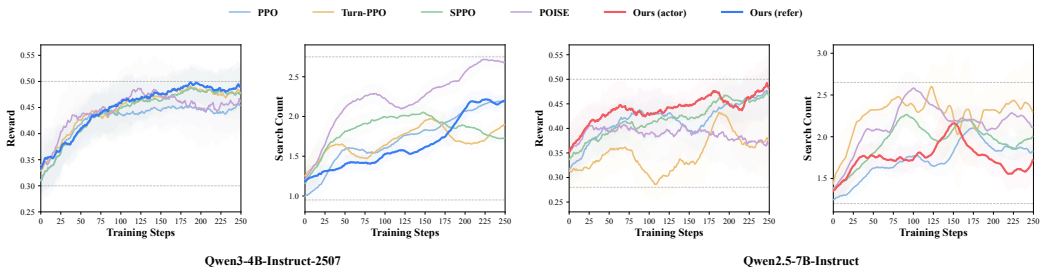

Figure 3: **Training dynamics of reward and search count.** We track reward and search count on Qwen3-4B-Instruct-2507 and Qwen2.5-7B-Instruct. GCM-PPO maintains strong reward while using fewer search actions than most baselines, reflecting concise interaction trajectories.

evaluate GCM-PPO on Qwen3-4B-Instruct-2507 and Qwen2.5-7B-Instruct in the main experiments, with additional results on smaller backbones reported in Appendix C. The task prompts are provided in Appendix G, while the computational resources are detailed in Appendix E.

4.2 Task Performance (RQ1)

We compare GCM-PPO against four critic-based and three critic-free methods. As shown in Table 1, GCM-PPO achieves the highest average performance among the evaluated methods on both backbones, with consistent gains across task complexity and out-of-domain evaluation.

Obs.1. Jointly matching credit granularity and critic capacity improves upon conventional value learning. On Qwen2.5-7B-Instruct, GCM-PPO achieves an average score of 0.404, outperforming PPO, Turn-PPO, SPPO, and POISE by 4.1, 6.3, 3.2, and 10.0 percentage points, respectively. On Qwen3-4B-Instruct-2507, it reaches 0.430, compared with 0.418 for Turn-PPO and 0.425 for SPPO. These gains highlight the importance of aligning credit-assignment granularity with critic capacity, rather than optimizing either dimension in isolation.

Obs.2. Learned turn-level credit outperforms group-relative optimization. GCM-PPO surpasses GRPO, GiGPO, and Tree-GRPO in average performance on both backbones. Notably, it improves upon Tree-GRPO, the strongest critic-free baseline, from 0.415 to 0.430 on Qwen3-4B-Instruct-2507 and from 0.390 to 0.404 on Qwen2.5-7B-Instruct. On the latter, the multi-hop average increases from 0.322 to 0.330. These improvements underscore the advantage of learning temporal credit at interaction boundaries over relying solely on group-relative trajectory comparisons.

4.3 Learning Efficiency (RQ2)

We examine the performance-cost tradeoff, reward and search dynamics, and representation-source choices of GCM-PPO. The results show that compact value estimation substantially reduces training cost while preserving effective temporal credit assignment and efficient search behavior.

Obs.3. Capacity-matched critics substantially improve the performance-cost tradeoff. As shown in Figure 2, GCM-PPO achieves an average score of 0.404 with a 30M-parameter critic in 282 minutes on Qwen2.5-7B-Instruct. Compared with PPO, it reduces critic parameters by approximately **237**$\times$ and training time by **54%**. It also outperforms SPPO while training 1.77$\times$ faster. On Qwen3-4B-Instruct-2507, GCM-PPO achieves the highest average score of 0.430 with only 15.7M critic parameters in 288 minutes. In contrast, POISE requires 488 minutes on Qwen2.5-7B-Instruct despite using only 7.2K critic parameters, demonstrating that critic size alone does not determine end-to-end efficiency. The compact critic of GCM-PPO additionally enables asynchronous CPU optimization to overlap with GPU-side actor optimization and rollout generation.

Obs.4. GCM-PPO improves reward without increasing search frequency. Figure 3 shows that GCM-PPO achieves rapid reward improvements while maintaining moderate search counts on Qwen2.5-7B-Instruct. Its search frequency decreases after the mid-training peak while reward continues to improve. On Qwen3-4B-Instruct-2507, the reference variant achieves higher reward than

Table 2: **Ablation study of GCM-PPO.** We evaluate the contributions of the temporal abstraction, linear branch, residual branch, and geometric turn ratio on Qwen3-4B-Instruct-2507 and Qwen2.5-7B-Instruct. † and ∗ denote in-domain and out-of-domain datasets, respectively.

Method	Single-Hop QA			Multi-Hop QA				Avg.
	NQ†	TriviaQA*	PopQA*	HotpotQA†	2Wiki*	MuSiQue*	Bamboogle*	
Qwen3-4B-Instruct-2507								
Ours (full)	0.436	0.647	0.440	0.405	0.430	0.177	0.476	0.430
w/o turn	0.402	0.639	0.434	0.408	0.422	0.187	0.490	0.426
w/o linear	0.436	0.623	0.437	0.390	0.351	0.192	0.387	0.402
w/o residual	0.440	0.638	0.445	0.415	0.385	0.184	0.468	0.425
w/o geometric	0.445	0.646	0.450	0.429	0.395	0.191	0.444	0.429
Qwen2.5-7B-Instruct								
Ours (full)	0.444	0.624	0.440	0.390	0.364	0.180	0.387	0.404
w/o turn	0.422	0.590	0.454	0.360	0.361	0.139	0.371	0.385
w/o linear	0.435	0.597	0.412	0.358	0.310	0.122	0.306	0.363
w/o residual	0.384	0.546	0.384	0.304	0.161	0.094	0.290	0.309
w/o geometric	0.439	0.618	0.427	0.400	0.299	0.170	0.363	0.388

PPO during late training while maintaining lower search counts throughout much of the early and middle training stages. These dynamics demonstrate that GCM-PPO improves the effectiveness of agent-environment interactions rather than relying on additional tool calls. Qualitative examples of agent interaction on Qwen2.5-7B-Instruct are provided in Appendix F.

Obs.5. Reused representations enable effective value learning with backbone-dependent source preferences. Both actor and reference representations support lightweight value estimation, with their relative performance depending on the backbone. Actor features achieve 0.404 versus 0.366 for reference features on Qwen2.5-7B-Instruct, whereas reference features achieve 0.430 versus 0.425 on Qwen3-4B-Instruct-2507. These results demonstrate the flexibility of reusing interaction-boundary representations for value learning without requiring a dedicated backbone.

4.4 Ablation Study (RQ3)

We ablate turn-level abstraction, the linear and residual branches of the value head, and geometric-mean importance ratios. Table 2 shows that all four components contribute to aggregate performance on both backbones, with the relative importance of the value-head branches varying across models.

Obs.6. Turn-level abstraction improves capacity-matched credit assignment. Reverting to token-level credit reduces average performance from 0.404 to 0.385 on Qwen2.5-7B-Instruct, with a larger decline on multi-hop QA (0.330 to 0.308) than on single-hop QA (0.503 to 0.489). On Qwen3-4B-Instruct-2507, removing turn-level abstraction also reduces the overall score from 0.430 to 0.426. These results highlight the importance of aligning value prediction with agent-environment transitions. By eliminating redundant within-turn prediction states while preserving inter-turn temporal credit, turn-level abstraction reduces the resolution required from a compact critic.

Obs.7. Linear decoding and nonlinear residual correction provide complementary value-learning capacity. On Qwen2.5-7B-Instruct, removing the linear branch reduces the average score from 0.404 to 0.363, while removing the residual branch causes a substantially larger decline to 0.309. The latter reduces multi-hop performance from 0.330 to 0.212. Interestingly, Qwen3-4B-Instruct-2507 exhibits the opposite sensitivity, with linear branch removal reducing performance from 0.430 to 0.402, compared with 0.425 when removing the residual branch. These findings highlight the complementary roles of the two branches: linear decoding extracts value information already encoded in LLM representations, while nonlinear residual correction provides additional capacity to capture value structure beyond direct feature decoding.

Obs.8. Geometric aggregation enables length-normalized turn-level policy updates. Replacing the geometric-mean importance ratio with the product of token-wise ratios lowers average performance from 0.404 to 0.388 on Qwen2.5-7B-Instruct and from 0.430 to 0.429 on Qwen3-4B-

Instruct-2507. Geometric aggregation normalizes log-likelihood changes by response length, reducing length sensitivity and ensuring consistent clipping across variable-length interactions.

5 Related Work

Multi-Turn LLM Agents. LLM agents extend language modeling to sequential decision making by interleaving reasoning, tool use, actions, and environment feedback over multi-turn trajectories (Yao et al., 2022b; Shinn et al., 2023; Schick et al., 2023; Qin et al., 2024; Li et al., 2023a; Zeng et al., 2024; Xi et al., 2025; Lu et al., 2026). Interactive benchmarks further expose agents to long-horizon web navigation and grounded decision making, where success depends on maintaining plans and adapting behavior across successive observations (Yao et al., 2022a; Zhou et al., 2024a; Deng et al., 2023; Liu et al., 2024; Drouin et al., 2024; Koh et al., 2024; Xie et al., 2024; Yao et al., 2024; Abdulhai et al., 2023). These settings motivate our focus on efficient reinforcement learning over turn-structured trajectories, where environment interactions provide natural temporal boundaries for credit assignment.

Credit Assignment in Agentic RL. Long-horizon agent training turns sparse or delayed outcomes into a temporal credit-assignment problem, motivating supervision at trajectory, sequence, segment, turn, and step granularity across hierarchical, group-relative, and structured rollout objectives (Feng et al., 2026b; He et al., 2026; Cheng et al., 2026; Ji et al., 2026; Li et al., 2026a; Wang et al., 2026; Zhou et al., 2024b; Guo et al., 2026; Li et al., 2026c; Liu et al., 2026b; Ding et al., 2026; Cao et al., 2026; Chen et al., 2026; Li et al., 2026b). Across these formulations, intermediate credit is obtained through shared-state comparison, structured rollout dependencies, intermediate reward signals, or temporal decomposition of interaction trajectories (Feng et al., 2026b; He et al., 2026; Cheng et al., 2026; Wei et al., 2025a; Ji et al., 2026; Li et al., 2026a; Zhou et al., 2024b; Xie et al., 2026; Wang et al., 2025; Qian et al., 2026). The challenge is pronounced in search, tool-use, and deep-research agents, where verifiable outcomes become available only after extended sequences of reasoning, retrieval, and environment interaction (Jin et al., 2025; Zheng et al., 2025; Wei et al., 2025a; Li et al., 2026a; Xie et al., 2026; Wang et al., 2025; Qi et al., 2025; Feng et al., 2026a; Wei et al., 2025b). Our work builds on this progression toward temporally localized credit and studies the complementary problem of matching credit granularity with compact value-estimator capacity.

Value Estimation in LLM RL. Critic-based policy optimization provides value estimates for variance reduction (Schulman et al., 2017; 2015), but applying actor-critic PPO to LLMs requires an additional value model and dense autoregressive value prediction, incurring substantial training and memory overhead (Stiennon et al., 2020; Ouyang et al., 2022; Li et al., 2026a; Wang et al., 2026). Recent work restructures value estimation around turn or sequence boundaries (Li et al., 2026a; Wang et al., 2026), decouples or pretrains value models (Huang et al., 2026), replaces online value estimation with sampled or offline targets (Brantley et al., 2026; Kazemnejad et al., 2024), extracts compact value signals from policy representations (Choi et al., 2026), or avoids learned critics through REINFORCE-style baselines (Ahmadian et al., 2024; Li et al., 2023b). AsyPPO (Liu et al., 2026a) employs lightweight mini-critics for value estimation and exploits inter-critic disagreement during policy optimization. SAPO (Liang et al., 2026) instead shares an autoregressive backbone between policy and value prediction and estimates values at causal interaction boundaries, reducing the overhead of maintaining a separate critic. GCM-PPO targets the efficiency bottleneck through turn-level abstraction, a compact value estimator, and asynchronous critic optimization.

6 Conclusion

We present GCM-PPO, an efficient framework for agentic RL built on the coupling between credit-assignment granularity and critic capacity. By combining turn-level credit assignment with adaptive lightweight value estimation and asynchronous critic optimization, GCM-PPO preserves decision-level learning signals while reducing computational and scheduling overhead. Experiments across multiple datasets and model scales show that GCM-PPO matches or outperforms strong baselines with substantially improved training efficiency. Our findings highlight the importance of jointly aligning credit granularity, critic capacity, and execution cost for efficient long-horizon agentic RL.

AI USE STATEMENT

Generative AI tools were used to assist with literature retrieval and discovery, as well as to improve the clarity, grammar, and presentation of the manuscript. OpenAI Codex was used to assist with the implementation and editing of research code. All relevant references and AI-assisted text were independently verified and reviewed by the authors. No generative AI tools were used for research ideation, methodology development, experimental design, or result analysis. The authors take full responsibility for the scientific contributions and final content of this paper.

ETHICS STATEMENT

This work aims to improve the training efficiency and credit assignment of LLM agents. While the proposed method does not directly address safety, more efficient agent training may facilitate the development of increasingly capable autonomous systems, potentially introducing risks of misuse or unintended behavior. Our experiments focus on research benchmarks and do not establish the safety or reliability of agents in real-world settings. Appropriate safety evaluations, access controls, and human oversight should therefore be considered before deployment.

REPRODUCIBILITY STATEMENT

We provide the algorithmic formulation of GCM-PPO and the experimental details in the main paper and appendix. To facilitate reproducibility, our anonymized code repository is available at https://anonymous.4open.science/r/GCM-PPO-C78D, including the implementation, training and evaluation configurations, and hyperparameters. These materials document the turn-level credit assignment mechanism, lightweight critic architecture, and asynchronous CPU–GPU execution, supporting reproduction of both the effectiveness and efficiency experiments.

REFERENCES

Marwa Abdulhai, Isadora White, Charlie Snell, Charles Sun, Joey Hong, Yuexiang Zhai, Kelvin Xu, and Sergey Levine. Lmrl gym: Benchmarks for multi-turn reinforcement learning with language models. *arXiv preprint arXiv:2311.18232*, 2023.

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. *arXiv preprint arXiv:2303.08774*, 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, 2024.

Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, OpenAI Pieter Abbeel, and Wojciech Zaremba. Hindsight experience replay. *Advances in neural information processing systems*, 30, 2017.

Jose A Arjona-Medina, Michael Gillhofer, Michael Widrich, Thomas Unterthiner, Johannes Brandstetter, and Sepp Hochreiter. Rudder: Return decomposition for delayed rewards. *Advances in Neural Information Processing Systems*, 32, 2019.

Kianté Brantley, Mingyu Chen, Zhaolin Gao, Jason Lee, Wen Sun, Wenhao Zhan, and Xuezhou Zhang. Accelerating rl for llm reasoning with optimal advantage regression. *Advances in Neural Information Processing Systems*, 38:151492–151531, 2026.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Ruike Cao, Shaojie Bai, Fugen Yao, Liang Dong, Jian Xu, and Li Xiao. Atpo: Adaptive tree policy optimization for multi-turn medical dialogue. *arXiv preprint arXiv:2603.02216*, 2026.

Yuhan Chen, Yuxuan Liu, Long Zhang, Pengzhi Gao, Jian Luan, and Wei Liu. Step: Success-rate-aware trajectory-efficient policy optimization. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 30681–30692, 2026.

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. *arXiv preprint arXiv:2605.07579*, 2026.

Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Sam Stevens, Boshi Wang, Huan Sun, and Yu Su. Mind2web: Towards a generalist agent for the web. *Advances in Neural Information Processing Systems*, 36:28091–28114, 2023.

Yifeng Ding, Hung Le, Songyang Han, Kangrui Ruan, Zhenghui Jin, Varun Kumar, Zijian Wang, and Anoop Deoras. Empowering multi-turn tool-integrated agentic reasoning with group turn policy optimization. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 42409–42423, 2026.

Alexandre Drouin, Maxime Gasse, Massimo Caccia, Issam H Laradji, Manuel Del Verme, Tom Marty, Léo Boisvert, Megh Thakkar, Quentin Cappart, David Vazquez, et al. Workarena: How capable are web agents at solving common knowledge work tasks? *arXiv preprint arXiv:2403.07718*, 2024.

Jiazhan Feng, Shijue Huang, Xingwei Qu, Ge Zhang, Yujia Qin, Baoquan Zhong, Chengquan Jiang, Jinxin Chi, and Wanjun Zhong. Retool: Reinforcement learning for strategic tool use in llms. In *International Conference on Learning Representations*, volume 2026, pp. 37909–37926, 2026a.

Lang Feng, Zhenghai Xue, Tingcong Liu, and Bo An. Group-in-group policy optimization for llm agent training. *Advances in Neural Information Processing Systems*, 38:46375–46408, 2026b.

Yiran Guo, Lijie Xu, Jie Liu, Dan Ye, and Shuang Qiu. Segment policy optimization: Effective segment-level credit assignment in rl for large language models. *Advances in Neural Information Processing Systems*, 38:114399–114431, 2026.

Shuo He, Lang Feng, Xin Cheng, Lei Feng, Bo An, et al. Hierarchy-of-groups policy optimization for long-horizon agentic tasks. In *International Conference on Learning Representations*, volume 2026, pp. 27572–27593, 2026.

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

Chenghua Huang, Lu Wang, Fangkai Yang, Pu Zhao, Qingwei Lin, Dongmei Zhang, and Saravan Rajmohan. Pretrain value, not reward: Decoupled value policy optimization. In *International Conference on Learning Representations*, volume 2026, pp. 115045–115066, 2026.

Yuxiang Ji, Ziyu Ma, Yong Wang, Guanhua Chen, Xiangxiang Chu, and Liaoni Wu. Tree search for llm agent reinforcement learning. In *International Conference on Learning Representations*, volume 2026, pp. 87362–87388, 2026.

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. *arXiv preprint arXiv:2503.09516*, 2025.

Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. 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)*, pp. 1601–1611, 2017.

Amirhossein Kazemnejad, Milad Aghajohari, Eva Portelance, Alessandro Sordoni, Siva Reddy, Aaron Courville, and Nicolas Le Roux. Vineppo: Refining credit assignment in rl training of llms. *arXiv preprint arXiv:2410.01679*, 2024.

Jing Yu Koh, Robert Lo, Lawrence Jang, Vikram Duvvur, Ming Lim, Po-Yu Huang, Graham Neubig, Shuyan Zhou, Russ Salakhutdinov, and Daniel Fried. Visualwebarena: Evaluating multimodal agents on realistic visual web tasks. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 881–905, 2024.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466, 2019.

Junbo Li, Peng Zhou, Rui Meng, Meet P Vadera, Lihong Li, and Yang Li. Turn-ppo: Turn-level advantage estimation with ppo for improved multi-turn rl in agentic llms. In *Findings of the Association for Computational Linguistics: EACL 2026*, pp. 6227–6243, 2026a.

Minghao Li, Yingxiu Zhao, Bowen Yu, Feifan Song, Hangyu Li, Haiyang Yu, Zhoujun Li, Fei Huang, and Yongbin Li. Api-bank: A comprehensive benchmark for tool-augmented llms. In *Proceedings of the 2023 conference on empirical methods in natural language processing*, pp. 3102–3116, 2023a.

Yu Li, Sizhe Tang, and Tian Lan. Reason in chains, learn in trees: Self-rectification and grafting for multi-turn agent policy optimization. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 4673–4686, 2026b.

Yugu Li, Zehong Cao, Jianglin Qiao, and Siyi Hu. Ssvpo: Effective step-level credit assignment for rl training of language models. In *The Fourteenth International Conference on Learning Representations*, 2026c.

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. *arXiv preprint arXiv:2310.10505*, 2023b.

Dayang Liang, Lang Feng, Bo An, and Yunlong Liu. Sapo: Single-rollout autoregressive policy optimization for agentic reinforcement learning. *arXiv preprint arXiv:2608.19842*, 2026.

Jiashun Liu, Johan S Obando Ceron, Han Lu, Yancheng He, Weixun Wang, Bo Zheng, Pablo Samuel Castro, Aaron Courville, Ling Pan, et al. Asymmetric proximal policy optimization: Mini-critics boost llm reasoning. In *International Conference on Learning Representations*, volume 2026, pp. 12574–12593, 2026a.

Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, et al. Agentbench: Evaluating llms as agents. In *International Conference on Learning Representations*, volume 2024, pp. 52989–53046, 2024.

Yuze Liu, Chaochao Lu, and Chao Yang. Shadow: Dynamic-aware credit assignment against long-horizon tasks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 23935–23944, 2026b.

Zhengxi Lu, Jiabo Ye, Fei Tang, Yongliang Shen, Haiyang Xu, Ziwei Zheng, Weiming Lu, Ming Yan, Fei Huang, Jun Xiao, et al. Experience-driven multi-turn reinforcement learning for gui agents. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9474–9496, 2026.

Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. 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)*, pp. 9802–9822, 2023.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35: 27730–27744, 2022.

Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A Smith, and Mike Lewis. Measuring and narrowing the compositionality gap in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 5687–5711, 2023.

Zehan Qi, Xiao Liu, Iat Long Iong, Hanyu Lai, Xueqiao Sun, Jiadai Sun, Xinyue Yang, Yu Yang, Shuntian Yao, Wei Xu, et al. Webrl: Training llm web agents via self-evolving online curriculum reinforcement learning. In *International Conference on Learning Representations*, volume 2025, pp. 79791–79821, 2025.

Cheng Qian, Emre Can Acikgoz, Qi He, Hongru Wang, Xiusi Chen, Dilek Hakkani-Tur, Gokhan Tur, and Heng Ji. Toolrl: Reward is all tool learning needs. *Advances in Neural Information Processing Systems*, 38:105523–105553, 2026.

Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, et al. Toolllm: Facilitating large language models to master 16000+ real-world apis. In *International Conference on Learning Representations*, volume 2024, pp. 9695–9717, 2024.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. *Advances in neural information processing systems*, 36:68539–68551, 2023.

John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. *Advances in neural information processing systems*, 36:8634–8652, 2023.

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. *Advances in neural information processing systems*, 33:3008–3021, 2020.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. MuSiQue: Multi-hop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554, 2022. doi: 10.1162/tacl_a_00475.

Tianyi Wang, Yixia Li, Long Li, Yibiao Chen, Shaohan Huang, Yun Chen, Peng Li, Yang Liu, and Guanhua Chen. Sppo: Sequence-level ppo for long-horizon reasoning tasks. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 7831–7853, 2026.

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. *arXiv preprint arXiv:2504.20073*, 2025.

Quan Wei, Siliang Zeng, Chenliang Li, William Brown, Oana Frunza, Wei Deng, Anderson Schneider, Yuriy Nevmyvaka, Yang Katie Zhao, Alfredo Garcia, et al. Reinforcing multi-turn reasoning in llm agents via turn-level reward design. *arXiv preprint arXiv:2505.11821*, 2025a.

Zhepei Wei, Wenlin Yao, Yao Liu, Weizhi Zhang, Qin Lu, Liang Qiu, Changlong Yu, Puyang Xu, Chao Zhang, Bing Yin, et al. Webagent-r1: Training web agents via end-to-end multi-turn reinforcement learning. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 7920–7939, 2025b.

Zhiheng Xi, Yiwen Ding, Wenxiang Chen, Boyang Hong, Honglin Guo, Junzhe Wang, Xin Guo, Dingwen Yang, Chenyang Liao, Wei He, et al. Agentgym: Evaluating and training large language model-based agents across diverse environments. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 27914–27961, 2025.

Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, Toh J Hua, Zhoujun Cheng, Dongchan Shin, Fangyu Lei, et al. Osworld: Benchmarking multimodal agents for open-ended tasks in real computer environments. *Advances in Neural Information Processing Systems*, 37:52040–52094, 2024.

Yutao Xie, Nathaniel Thomas, Nick Hansen, Yang Fu, Li Li, and Xiaolong Wang. Tips: Turn-level information-potential reward shaping for search-augmented llms. In *International Conference on Learning Representations*, volume 2026, pp. 156549–156584, 2026.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D Manning. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pp. 2369–2380, 2018.

Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. Webshop: Towards scalable real-world web interaction with grounded language agents. *Advances in Neural Information Processing Systems*, 35:20744–20757, 2022a.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*, 2022b.

Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik Narasimhan. τ-bench: A benchmark for tool-agent-user interaction in real-world domains. *arXiv preprint arXiv:2406.12045*, 2024.

Aohan Zeng, Mingdao Liu, Rui Lu, Bowen Wang, Xiao Liu, Yuxiao Dong, and Jie Tang. Agenttuning: Enabling generalized agent abilities for llms. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 3053–3077, 2024.

Chenchen Zhang. From reasoning to agentic: Credit assignment in reinforcement learning for large language models. *arXiv preprint arXiv:2604.09459*, 2026.

Guibin Zhang, Hejia Geng, Xiaohang Yu, Zhenfei Yin, Zaibin Zhang, Zelin Tan, Heng Zhou, Zhongzhi Li, Xiangyuan Xue, Yijiang Li, et al. The landscape of agentic reinforcement learning for llms: A survey. *arXiv preprint arXiv:2509.02547*, 2025.

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, 2025.

Shuyan Zhou, Frank F Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, et al. Webarena: A realistic web environment for building autonomous agents. In *International Conference on Learning Representations*, volume 2024, pp. 15585–15606, 2024a.

Yifei Zhou, Andrea Zanette, Jiayi Pan, Sergey Levine, and Aviral Kumar. Archer: Training language model agents via hierarchical multi-turn rl. *arXiv preprint arXiv:2402.19446*, 2024b.

A Frequently Asked Questions (FAQs)

This section clarifies the central design choices of GCM-PPO, focusing on the interaction among temporal abstraction, value-function capacity, and execution efficiency.

A.1 Q1: Do the experiments isolate the interaction between credit granularity and critic capacity?

Yes. PPO, Turn-PPO, GCM-PPO without turn-level abstraction, and full GCM-PPO form a 2×2 comparison over token-level versus turn-level credit assignment and actor-scale versus compact critic capacity. PPO combines token-level credit with an actor-scale critic, Turn-PPO combines turn-level credit with an actor-scale critic, the w/o turn variant combines token-level credit with our compact critic, and full GCM-PPO combines turn-level credit with the compact critic. Across both backbones, the turn-level compact configuration achieves the highest average performance among these four settings, reaching 0.430 on Qwen3-4B-Instruct-2507 and 0.404 on Qwen2.5-7B-Instruct, the best results among all evaluated methods. This factorial comparison provides direct empirical evidence for granularity-capacity coupling. Future work will extend this analysis with broader sweeps over temporal granularity and critic capacity to characterize their interaction.

A.2 Q2: How does GCM-PPO differ from closely related efficient policy-optimization methods?

Existing efficient policy-optimization methods primarily reduce cost through temporal abstraction or value-model redesign. Turn-level formulations align credit with interaction boundaries while retaining a model-scale critic. Mini-critic ensembles reduce value-model capacity through architectural asymmetry and incorporate critic disagreement into policy optimization. Sequence-level methods assign a prompt-conditioned advantage to the entire response, eliminating inter-turn credit. Internal-state probes derive trajectory-level value signals from paired rollouts, while shared-backbone actor-critic methods integrate policy and value prediction within the same autoregressive model.

GCM-PPO jointly coordinates temporal abstraction, critic capacity, and execution efficiency within one framework. It preserves single-rollout turn-level GAE, maps detached LLM representations to values through a compact linear-plus-residual critic, and asynchronously optimizes the critic on CPU alongside GPU-side actor optimization and rollout generation. This design retains inter-turn temporal credit while reducing value-model capacity and sequential critic cost.

A.3 Q3: If critic capacity is governed by temporal granularity, why does the capacity rule depend on actor scale?

Temporal granularity determines the complexity of the value-prediction target, while actor scale and runtime determine the critic capacity that can be allocated efficiently. Token-level prediction must distinguish densely adjacent prefixes without intervening environment feedback, trajectory-level prediction removes intermediate temporal structure, and turn-level prediction retains states separated by complete actions and new observations. Within the turn-level regime, actor scale determines the available GPU-side overlap window and therefore constrains the residual width that can be hidden from the critical path. The scale rule selects critic capacity within this execution budget, preserving a compact value model while adapting its residual capacity across actor scales.

A.4 Q4: Does turn-level abstraction discard important within-turn credit?

Turn-level abstraction removes token-prefix credit distinctions within a response, but preserves the decision-relevant credit associated with environment-level actions. An agent turn constitutes the action presented to the environment, and its tokens are generated without intervening observations, so intermediate prefixes do not correspond to independent environment-level decisions. GCM-PPO therefore assigns one GAE advantage to the complete turn and optimizes all generated tokens through a length-normalized turn likelihood ratio. This retains token-level differentiability while aligning credit assignment with the interaction structure. When feedback occurs during generation, each feedback event defines a new interaction boundary under the same formulation.

Algorithm 1 GCM-PPO: Unified Turn-Level Optimization with Actor/Reference Features

Require: Actor π_θ; residual critic C_ϕ; source $\texttt{src} \in \{\texttt{actor}, \texttt{refer}\}$; optional frozen reference π_{ref}
Require: PPO parameters $\gamma, \lambda, \epsilon, \epsilon_v$
Ensure: Updated parameters θ, ϕ

for each training iteration **do**
  $\theta_{\text{old}} \leftarrow \theta$
  $\mathcal{B} \leftarrow \textsc{Rollout}(\pi_{\theta_{\text{old}}})$ ▷ Sample actor trajectories
  $(\mathcal{S}, \mathcal{A}, \mathcal{R}, \mathbf{d}, \mathcal{I}) \leftarrow \textsc{ExtractTurns}(\mathcal{B})$ ▷ Turn transitions and response spans
  $(\ell_{\text{old}}, H_{\text{actor}}) \leftarrow \textsc{ActorForward}(\pi_{\theta_{\text{old}}}, \mathcal{B})$
  if $\texttt{src} = \texttt{actor}$ **then**
    $H \leftarrow \textsc{StopGradient}(H_{\text{actor}})$
  else
    $H_{\text{refer}} \leftarrow \textsc{ReferenceForward}(\pi_{\text{ref}}, \mathcal{B})$
    $H \leftarrow \textsc{StopGradient}(H_{\text{refer}})$
  end if
  $\textsc{WaitForPreviousValueUpdate}$ ▷ Complete the previous critic update
  $V_{\text{old}} \leftarrow \textsc{PredictTurnValues}(C_\phi, H, \mathcal{I})$ ▷ Evaluate pre-turn states
  $(\widehat{A}, \widehat{R}) \leftarrow \textsc{TurnGAE}(\mathcal{R}, V_{\text{old}}, \mathbf{d}, \mathcal{I}, \gamma, \lambda)$ ▷ Turn-level advantages and targets
  asynchronously on CPU: $\textsc{UpdateValueHead}(C_\phi, H, V_{\text{old}}, \widehat{R}, \mathcal{I}, \epsilon_v)$
  for each PPO minibatch $\mathcal{M} \subset \mathcal{B}$ **do**
    $\rho_{\mathcal{M}} \leftarrow \textsc{GeometricMeanTurnRatios}(\pi_\theta, \ell_{\text{old},\mathcal{M}}, \mathcal{M}, \mathcal{I}_{\mathcal{M}})$
    $\theta \leftarrow \textsc{ClippedPPOUpdate}(\theta, \rho_{\mathcal{M}}, \widehat{A}_{\mathcal{M}}, \epsilon)$ ▷ Mean clipped policy loss over turns
  end for
  $\textsc{SynchronizeRolloutPolicy}(\pi_\theta)$ ▷ Prepare the next rollout
end for
$\textsc{WaitForPreviousValueUpdate}$ ▷ Finish pending critic optimization
return θ, ϕ

A.5 Q5: Why can a compact residual critic replace a model-scale value network?

The reused boundary representation encodes the interaction history, reducing value learning to a mapping from a state representation to expected return. The linear branch extracts value information accessible in the LLM feature space, while the residual branch captures value-specific nonlinear structure. Detaching the representation confines optimization to this compact mapping and leaves the policy backbone unchanged. Actor features provide policy-adaptive representations, while reference features provide stationary representations. Both variants share the same value architecture and scale rule, making critic capacity explicit without duplicating the representation backbone.

A.6 Q6: Does asynchronous critic optimization introduce stale or inconsistent advantages?

Each batch is evaluated by a single frozen critic snapshot, ensuring consistent advantages and returns within one policy update. At iteration k, V_{ϕ_k} evaluates trajectories collected under π_{θ_k} and defines the fixed advantages and returns used for optimization. The update from ϕ_k to ϕ_{k+1} proceeds concurrently with actor optimization and generation of the next rollout batch, and completes before value prediction for that batch. Asynchronous execution therefore changes the computation schedule without altering the iteration-level dependency structure. Selecting critic width to fit within the actor-update and rollout window converts reduced value capacity into lower sequential training cost.

B Method Details

This section specifies the implementation of GCM-PPO, including the construction of turn-level transitions, the deterministic capacity rule for the residual critic, and the parameter dependencies induced by asynchronous optimization. The complete end-to-end training procedure for efficient

policy optimization is summarized in Algorithm 1. Ours (actor) reuses representations from the evolving policy, while Ours (refer) uses representations from a frozen reference model.

B.1 Capacity-Matched Temporal Abstraction

We construct turn-level transitions for each interaction trajectory from the alternating sequence of model responses and environment observations. If the tokens generated at turn t occupy the response-space interval $[b_t, e_t)$, this interval defines the complete action a_t. The subsequent environment observation remains in the autoregressive context and contributes to s_{t+1}, while its tokens are excluded from the action span. The state value $V(s_t)$ is aligned with the causal representation at position $b_t - 1$, immediately before the first generated token. This position is the final prompt token for the first turn and the final context token preceding each subsequent response.

Trajectory feedback is projected onto this turn structure before GAE. The terminal reward is assigned to the final valid model action. If a trajectory terminates after an environment observation, the reward remains associated with the preceding action. More generally, token-wise shaping rewards within $[b_t, e_t)$ are summed to form r_t, and the terminal bootstrap value is set to zero. In our setting, the KL reward coefficient is zero and $\gamma = \lambda = 1$, reducing the unnormalized targets to

$$\hat{R}_t = R_\tau, \qquad \hat{A}_t = R_\tau - V_{\bar{\phi}}(s_t).$$

Thus, visited turns share the same outcome target, while the state value provides turn-specific credit.

A sparse turn mask selects b_t for value supervision and advantage normalization. GAE produces one advantage and return per valid turn, and normalization statistics are computed over these quantities. Old-policy probabilities are recomputed on the exact token IDs sampled during rollout and grouped by the recorded action spans to form the geometric-mean ratio. Consistent with the main objective, the clipped terms are averaged over the global set $\mathcal{B}_{\text{turn}}$, with one contribution from each valid turn. We use one trajectory per prompt, policy clipping $\epsilon = 0.2$, and value clipping $\epsilon_v = 0.5$.

B.2 Adaptive Residual Value Estimation

The reused feature h_t is extracted from the transformer representation immediately before the language-model output projection at boundary position $b_t - 1$. Ours (actor) captures this feature during the pre-update actor forward pass used to evaluate old-policy probabilities, yielding a representation space that evolves with the policy. Ours (refer) extracts the feature from a frozen reference-model forward pass, yielding a stationary representation space across RL iterations. Both variants detach the boundary features and expose a d-dimensional interface to the residual critic.

We instantiate the critic using an explicit scale rule. For representation dimension d and residual width m, RMSNorm contributes d parameters, the linear branch contributes $d + 1$, and the residual branch contributes $dm + m^2 + 3m + 1$. The total trainable capacity is

$$N(m) = m^2 + (d + 3)m + 2d + 2.$$

Given actor parameter count P_π and capacity rate c, we define

$$M(P_\pi) = \max\left\{N(1), \text{round}\left(\frac{cP_\pi}{10^9}\right)\right\}, \qquad g(P_\pi) = \max\left\{m \in 64\mathbb{N} : N(m) \le M(P_\pi)\right\}.$$

We use $c = 4 \times 10^6$, corresponding to a critic budget of approximately 0.4% of the actor. It yields $m = 2624$ for Qwen2.5-3B-Instruct, $m = 2304$ for Llama-3.2-3B-Instruct, $m = 2880$ for Qwen3-4B-Instruct-2507, and $m = 3968$ for Qwen2.5-7B-Instruct. Because dm couples residual width to representation dimension, actor scale and hidden width jointly determine the instantiated critic.

The linear branch and the output projection of the residual branch are initialized to zero, giving $V_{\phi_0}(s) = 0$ at the start of RL training, so both branches initially contribute zero. Subsequent updates jointly learn the direct linear decoder and nonlinear residual correction. Value clipping is anchored at the predictions used to construct the current advantages, aligning the critic target, clipping center, and actor update with the same pre-update value snapshot.

Table 3: **Additional Search QA results on smaller backbones.** We extend the main comparison to Qwen2.5-3B-Instruct and Llama-3.2-3B-Instruct across single-hop and multi-hop QA benchmarks. † and $*$ denote in-domain and out-of-domain datasets, respectively.

Method	Single-Hop QA			Multi-Hop QA				Avg.
	NQ†	TriviaQA*	PopQA*	HotpotQA†	2Wiki*	MuSiQue*	Bamboogle*	
Qwen2.5-3B-Instruct								
PPO	0.401	0.570	0.404	**0.332**	0.321	**0.118**	**0.290**	**0.348**
GRPO	0.378	0.555	0.389	0.308	0.277	0.106	0.266	0.326
GiGPO	0.374	0.571	0.376	0.307	0.278	0.100	0.202	0.315
Tree-GRPO	0.369	0.547	0.368	0.307	0.294	0.116	0.250	0.322
Turn-PPO	0.361	0.511	0.263	0.253	0.249	0.046	0.129	0.259
SPPO	0.350	0.527	0.335	0.287	0.229	0.106	0.210	0.292
POISE	0.221	0.419	0.285	0.243	0.193	0.055	0.153	0.224
Ours (actor)	0.398	0.582	0.410	0.328	**0.336**	0.112	0.202	0.338
Ours (refer)	**0.411**	**0.593**	**0.419**	0.321	0.323	0.098	0.266	0.347
Llama-3.2-3B-Instruct								
PPO	**0.424**	0.585	**0.438**	**0.355**	**0.309**	**0.127**	0.266	**0.358**
GRPO	0.406	0.587	0.402	0.326	0.238	0.118	0.306	0.340
GiGPO	0.392	0.603	0.379	0.308	0.256	0.091	0.315	0.335
Tree-GRPO	0.391	0.592	0.377	0.328	0.279	0.120	**0.379**	0.352
Turn-PPO	0.276	0.501	0.197	0.199	0.270	0.042	0.169	0.236
SPPO	0.384	0.581	0.393	0.313	0.271	0.105	0.347	0.342
POISE	0.331	0.505	0.339	0.210	0.151	0.046	0.177	0.251
Ours (actor)	0.417	**0.619**	0.414	0.298	0.274	0.078	0.218	0.331
Ours (refer)	0.391	0.577	0.395	0.286	0.210	0.061	0.258	0.311

B.3 Asynchronous Lightweight Critic Optimization

Let $\mathcal{D}_k$ denote trajectories generated by π_{θ_k}, and let ϕ_k denote the critic obtained from the update on $\mathcal{D}_{k-1}$. Processing $\mathcal{D}_k$ begins with an actor or reference forward pass that produces the detached boundary-feature batch H_k. The training process then joins the pending critic update, evaluates V_{ϕ_k} on H_k, and constructs $\hat{A}_k$ and $\hat{R}_k$. These quantities define both the actor objective and the clipped value objective at iteration k.

Once the targets are fixed, a CPU worker updates the critic on $(H_k, V_{\phi_k}, \hat{R}_k)$ to obtain ϕ_{k+1}. Concurrently, the GPU updates θ_k using $\hat{A}_k$, transfers θ_{k+1} to the rollout engine, and generates $\mathcal{D}_{k+1}$. The two streams operate on disjoint parameter sets and share immutable feature, value, advantage, and return tensors. For Ours (actor), H_k remains the pre-update representation associated with θ_k throughout the CPU update. For Ours (refer), all H_k remain in the fixed reference feature space.

The next synchronization occurs immediately before value prediction on $\mathcal{D}_{k+1}$. The resulting sequence is V_{ϕ_k} for $\mathcal{D}_k$, followed by the update $\phi_k \to \phi_{k+1}$, then $V_{\phi_{k+1}}$ for $\mathcal{D}_{k+1}$. Critic width is fixed before RL training, and the capacity rate is selected according to

$$T_{\text{critic}}\left(g(P_\pi)\right) \leq T_{\text{actor}}(P_\pi) + T_{\text{rollout}}(P_\pi).$$

This schedule places critic optimization within the actor-update and rollout interval, while value prediction remains the synchronization boundary between consecutive critic versions.

C Additional Experiments

We extend our evaluation to Qwen2.5-3B-Instruct and Llama-3.2-3B-Instruct to investigate two research questions. **RQ4 (Smaller Backbones):** How does GCM-PPO perform on smaller backbones in terms of performance, training efficiency, and interaction dynamics? **RQ5 (Component Sensitivity):** How do the contributions of components vary across smaller backbones and QA benchmarks?

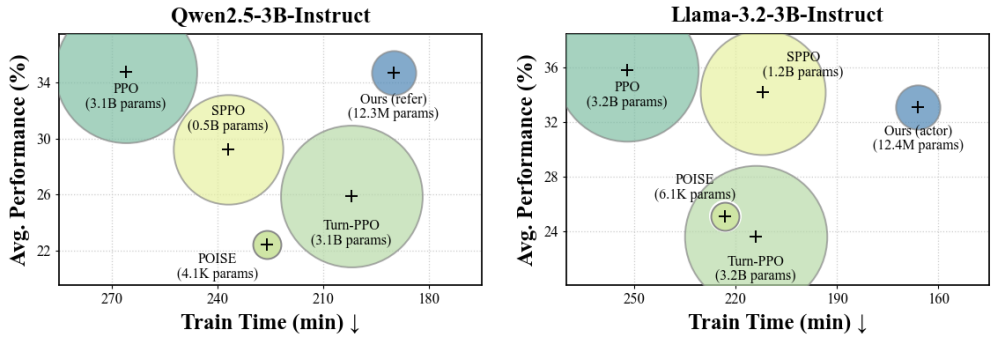

Figure 4: **Additional performance-efficiency comparison.** Bubble size represents critic parameter count, while the two axes report average QA performance and training time on smaller backbones.

C.1 Smaller Backbones (RQ4)

We extend the comparison to two 3B backbones. Table 3 reports performance across seven QA benchmarks, while Figures 4 and 5 present the performance-efficiency and training dynamics.

Obs.9. Smaller backbones exhibit distinct performance gains across task complexity. On Qwen2.5-3B-Instruct, GCM-PPO achieves an average score of 0.347, comparable to PPO's 0.348, while improving the single-hop average from 0.458 to 0.474. It achieves the highest accuracy on all three single-hop benchmarks. On Llama-3.2-3B-Instruct, GCM-PPO improves TriviaQA accuracy from 0.585 to 0.619, while PPO retains a higher overall average of 0.358 versus 0.331. The performance differences are more pronounced on multi-hop QA, where GCM-PPO achieves 0.217 compared with PPO's 0.264. These results reveal a stronger separation between single-hop retrieval and multi-hop evidence aggregation on smaller backbones.

Obs.10. Training-time reductions persist across smaller backbones despite different accuracy profiles. As shown in Figure 4, GCM-PPO achieves an average score of 0.347 in 190 minutes on Qwen2.5-3B-Instruct, compared with 0.348 in 266 minutes for PPO. On Llama-3.2-3B-Instruct, it completes training in 166 minutes, compared with 252 minutes for PPO, with average scores of 0.331 and 0.358, respectively. Both configurations use approximately 12M critic parameters. These results further show that substantial training-time reductions extend to smaller actor scales, while the corresponding accuracy tradeoffs depend on the particular backbone.

Obs.11. Smaller backbones exhibit different reward-search dynamics. Figure 5 shows distinct interaction patterns across the two 3B backbones. On Qwen2.5-3B-Instruct, the reference variant initially increases its search frequency before reducing tool use during later training, while reward continues to improve. On Llama-3.2-3B-Instruct, the actor variant maintains a comparatively high search frequency during intermediate and late training, whereas PPO achieves higher reward with fewer searches. These contrasting trajectories indicate that the relationship between search allocation and reward progression varies substantially across backbone families.

C.2 Component Sensitivity (RQ5)

We extend the ablation study to both 3B backbones in Table 4. The results reveal differences in component sensitivity across model scales and heterogeneous effects across individual benchmarks.

Obs.12. The contribution of turn-level abstraction depends on backbone and task complexity. Removing turn-level abstraction reduces average performance from 0.347 to 0.336 on Qwen2.5-3B-Instruct, with consistent declines across most benchmarks. On Llama-3.2-3B-Instruct, the average changes only slightly from 0.331 to 0.330, but individual benchmarks exhibit larger differences. In particular, removing turn-level abstraction increases HotpotQA accuracy from 0.298 to 0.327 while reducing 2Wiki accuracy from 0.274 to 0.238. These results show that the contribution of temporal abstraction varies across backbone representations and multi-hop reasoning tasks.

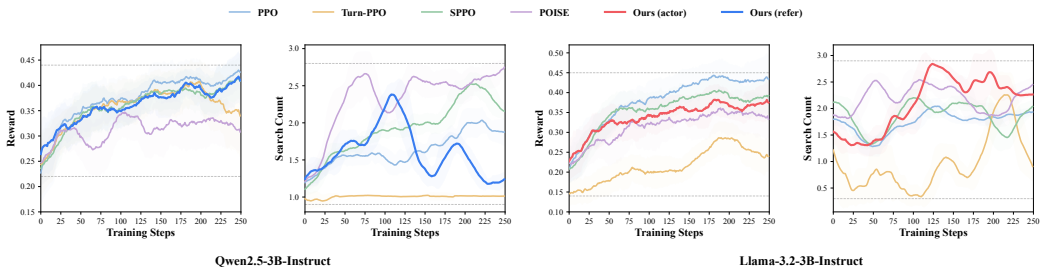

Figure 5: **Additional training dynamics of reward and search count.** We track reward and search count on Qwen2.5-3B-Instruct and Llama-3.2-3B-Instruct, extending the analysis to smaller backbones and revealing distinct optimization and interaction dynamics across methods.

Obs.13. Residual value correction exhibits consistent importance across 3B backbones. On Qwen2.5-3B-Instruct, removing the residual branch reduces average performance from 0.347 to 0.309, compared with 0.342 when removing the linear branch. On Llama-3.2-3B-Instruct, the scores are 0.319 and 0.329, compared with 0.331 for the full model. The residual branch has a pronounced effect on single-hop QA, with NQ accuracy decreasing by 10.5 and 5.4 percentage points on the two backbones, respectively. Unlike Qwen3-4B-Instruct-2507, where the linear branch has a larger aggregate contribution, both 3B backbones exhibit greater sensitivity to residual removal. This highlights the changing capacity requirements of lightweight value estimation across actor scales.

Obs.14. Importance-ratio aggregation induces heterogeneous benchmark-level effects. Replacing geometric-mean importance ratios with raw products produces small changes on both 3B backbones, reducing average performance from 0.347 to 0.344 on Qwen2.5-3B-Instruct and from 0.331 to 0.330 on Llama-3.2-3B-Instruct. However, the effects differ substantially across datasets. On Qwen2.5-3B-Instruct, removing geometric aggregation improves HotpotQA and MuSiQue accuracy by 3.5 and 4.3 percentage points, respectively, while reducing PopQA and Bamboogle accuracy by 3.5 and 3.2 points. On Llama-3.2-3B-Instruct, the modification improves Bamboogle accuracy but reduces performance on TriviaQA, HotpotQA, and 2Wiki. These results reveal task-dependent tradeoffs in importance-ratio aggregation that are not captured by aggregate performance.

D Implementation Details

This section describes the experimental protocol. We first detail QA datasets, including their construction, processed split sizes, and the capability emphasized by each benchmark. We then summarize seven baselines according to their optimization mechanism and credit-assignment granularity. Finally, we describe the normalized exact-match metric and answer normalization for evaluation.

D.1 Dataset Details

Based on reasoning structure, we group NQ, TriviaQA, and PopQA as single-hop QA, and HotpotQA, 2WikiMultiHopQA, MuSiQue, and Bamboogle as multi-hop QA. Following Search-R1, we use NQ and HotpotQA for training and sample 10,000 examples from the 169,615-example training pool to reduce training cost and accelerate iteration. All seven datasets are used for evaluation, with NQ and HotpotQA as in-domain datasets and the remaining five as out-of-domain datasets.

NQ (Kwiatkowski et al., 2019). Natural Questions uses anonymized Google Search queries, with each query paired with a Wikipedia page annotated with long and short answers. The processed release contains 79,168 training examples and 3,610 test examples. NQ evaluates retrieval and answer extraction for naturally occurring search queries across diverse information needs.

TriviaQA (Joshi et al., 2017). TriviaQA contains question-answer pairs authored by trivia enthusiasts, with supporting evidence independently collected from Wikipedia and the Web. The processed release contains 78,785 training examples and 11,313 test examples. TriviaQA emphasizes retrieval under substantial lexical and syntactic variation between questions and supporting evidence.

Table 4: **Additional ablation results on smaller backbones.** We evaluate the same component removals on Qwen2.5-3B-Instruct and Llama-3.2-3B-Instruct. The residual branch contributes consistently across both backbones. † and ∗ denote in-domain and out-of-domain datasets, respectively.

Method	Single-Hop QA			Multi-Hop QA				Avg.
	NQ†	TriviaQA*	PopQA*	HotpotQA†	2Wiki*	MuSiQue*	Bamboogle*	
Qwen2.5-3B-Instruct								
Ours (full)	0.411	0.593	0.419	0.321	0.323	0.098	0.266	0.347
w/o turn	0.389	0.575	0.403	0.307	0.298	0.096	0.282	0.336
w/o linear	0.370	0.561	0.424	0.316	0.314	0.104	0.306	0.342
w/o residual	0.306	0.545	0.397	0.299	0.269	0.092	0.258	0.309
w/o geometric	0.400	0.573	0.384	0.356	0.317	0.141	0.234	0.344
Llama-3.2-3B-Instruct								
Ours (full)	0.417	0.619	0.414	0.298	0.274	0.078	0.218	0.331
w/o turn	0.417	0.609	0.419	0.327	0.238	0.080	0.218	0.330
w/o linear	0.387	0.581	0.404	0.317	0.271	0.101	0.242	0.329
w/o residual	0.363	0.555	0.388	0.292	0.254	0.090	0.290	0.319
w/o geometric	0.405	0.594	0.417	0.280	0.260	0.089	0.265	0.330

PopQA (Mallen et al., 2023). PopQA is constructed from Wikidata subject-relation-object triples verbalized with relation-specific question templates, with entity popularity estimated from Wikipedia page views. The processed release provides 14,267 test examples for evaluation. PopQA evaluates factual retrieval over long-tail entities and sensitivity to variations in entity popularity.

HotpotQA (Yang et al., 2018). HotpotQA contains questions constructed from multiple Wikipedia documents and provides sentence-level supporting facts, including bridge and comparison questions. The processed release contains 90,447 training examples and 7,405 development examples. We use the development split for evaluation. HotpotQA evaluates cross-document evidence integration with explicit supporting-fact supervision.

2WikiMultiHopQA (Ho et al., 2020). 2WikiMultiHopQA combines Wikidata relations with Wikipedia text and constructs questions using controlled templates and rules. Each example provides evidence describing its reasoning chain. The processed release contains 15,000 training examples and 12,576 development examples. We use the development split for evaluation. 2WikiMultiHopQA emphasizes multi-step reasoning-path recovery across linked evidence.

MuSiQue (Trivedi et al., 2022). MuSiQue is built by composing connected single-hop questions, with later steps depending on earlier intermediate answers. Its filtering procedure enforces connected reasoning chains across two to four hops. The processed release contains 19,938 training examples and 2,417 development examples. We use the development split for evaluation. MuSiQue evaluates compositional reasoning with dependencies between intermediate steps.

Bamboogle (Press et al., 2023). Bamboogle contains manually written two-hop questions derived from Wikipedia vital articles and selected using failures of a popular search engine on the corresponding queries. The processed release provides 125 test examples for evaluation. Bamboogle evaluates decomposition and evidence composition on a compact set of challenging questions.

D.2 Baseline Details

We compare against seven baselines spanning three directions in policy optimization. PPO and GRPO represent foundational paradigms based on learned and group-relative baselines. GiGPO and Tree-GRPO extend group-relative optimization with finer-grained credit assignment for multi-turn agents. Turn-PPO, SPPO, and POISE explore efficient learned value estimation across different temporal granularities and representation sources.

PPO (Schulman et al., 2017). PPO alternates between on-policy rollout collection and clipped policy optimization. In the actor-critic formulation, a critic estimates state values, generalized advantage estimation combines rewards and values into advantages, and the actor is optimized with a

clipped importance-ratio objective while the critic fits return targets. In our implementation, value estimation and credit assignment operate at token level throughout policy optimization.

GRPO (Shao et al., 2024). GRPO samples multiple responses for each prompt and computes relative advantages by normalizing rewards within the group. These group-relative advantages are optimized with a PPO-style clipped objective and reference-policy regularization, providing the learning signal across sampled responses for each subsequent policy update.

GiGPO (Feng et al., 2026b). GiGPO augments group-relative optimization with hierarchical advantages at episode and step levels. It derives macro relative advantages from complete-trajectory returns, identifies repeated environment states as anchors, and groups actions from each anchor to compute micro relative advantages from discounted returns. The two signals jointly provide localized credit for individual decisions in multi-turn optimization. We define states using the initial task and subsequent retrieved observations. State occurrences within the same question are greedily clustered against the first representative using character-level similarity with threshold 0.9. Local returns are $\gamma^{T-1-t}R$ with $\gamma = 0.95$, standardized within each cluster, and added to the trajectory-level advantage with unit weight. Singleton clusters receive zero local advantage.

Tree-GRPO (Ji et al., 2026). Tree-GRPO organizes rollouts as a tree whose nodes represent complete interaction steps and whose branches share common prefixes. Outcome rewards are propagated through the tree to construct intra-tree and inter-tree relative advantages, yielding step-level process signals for group-relative optimization. We use two initial trees and expand two sampled states per tree for one round, producing six leaves per prompt. Terminal rewards are standardized globally across all leaves and locally within each three-leaf tree. The two normalized signals are summed to define each leaf advantage used in the policy update.

Turn-PPO (Li et al., 2026a). Turn-PPO formulates agent learning as a turn-level Markov decision process. Each model response forms a turn-level action, environment feedback defines the next state, and value prediction and generalized advantage estimation operate over turn-level transitions. The advantages support PPO updates at interaction-turn granularity. Our implementation retains an actor-scale critic while moving value estimation, GAE, and clipping to turn boundaries. The turn ratio is the geometric mean of token-wise likelihood ratios within each response.

SPPO (Wang et al., 2026). SPPO formulates long-horizon reasoning as a sequence-level contextual bandit. It treats the complete generated sequence as a single action and learns a decoupled scalar value function as an outcome-level baseline for each trajectory. The resulting sequence-level advantage is optimized with a PPO-style clipped objective. The critic processes only the initial prompt and predicts success from its final valid token. We use Qwen2.5-1.5B-Instruct, Qwen2.5-0.5B-Instruct, Qwen3-0.6B, and Llama-3.2-1B-Instruct as critics for Qwen2.5-7B-Instruct, Qwen2.5-3B-Instruct, Qwen3-4B-Instruct-2507, and Llama-3.2-3B-Instruct, respectively. The critic backbone and scalar head are optimized with binary outcome supervision during RL training.

POISE (Choi et al., 2026). POISE estimates the baseline from internal signals produced during the policy forward pass for each trajectory. A lightweight probe consumes hidden representations from the prompt and generated trajectory together with token-entropy statistics. Its cross-rollout construction uses an independent rollout to estimate the baseline from policy representations. We sample two trajectories per prompt and predict each baseline from its partner's features. The feature vector concatenates pooled prompt and generated-token representations from Transformer block 19 with the mean, standard deviation, and maximum token entropy. The Ridge probe is updated after actor optimization, so current advantages use parameters fitted on preceding rollouts.

D.3 Evaluation Metrics

We report normalized exact-match (EM) accuracy for all QA datasets following the Search-R1 evaluation protocol. We extract the final answer span enclosed by `<answer>` and `</answer>` from the model output. The prediction and each reference answer are normalized by lowercasing, removing punctuation, deleting the English articles `a`, `an`, and `the`, and collapsing consecutive whitespace. Each example receives a score of 1 when the normalized prediction exactly matches any normalized reference answer and 0 otherwise. The reported EM accuracy is the average of these binary scores over all examples in the corresponding evaluation split.

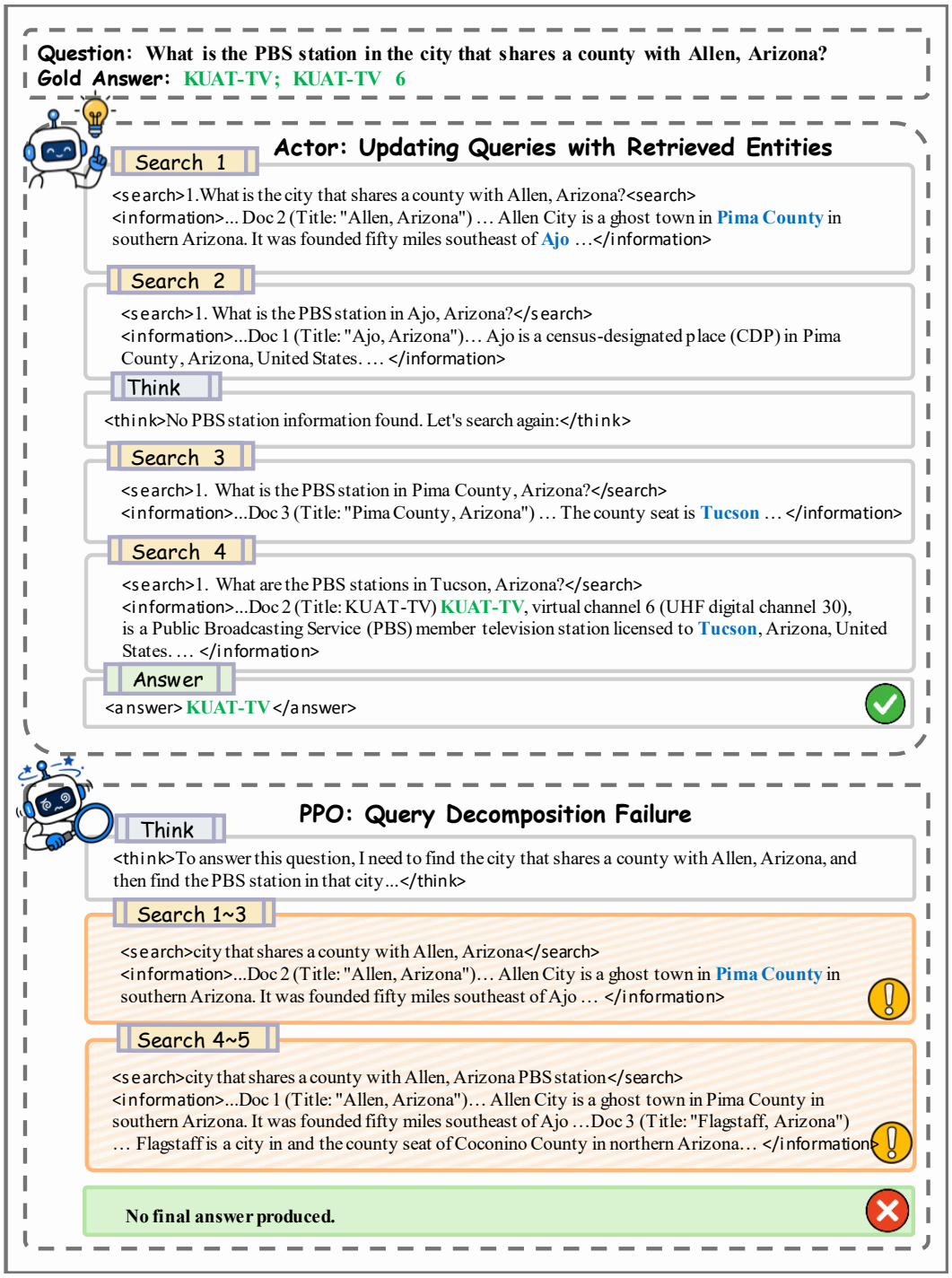

Figure 6: **Query decomposition failure.** PPO issues the compound query and fails to convert retrieved intermediate entities into search targets, producing no final answer. Ours (actor) updates the query with retrieved entities and reaches the target PBS station, KUAT-TV.

E Computational Resources

We implemented GCM-PPO using Slime v0.3.0, an LLM reinforcement learning framework that integrates Megatron-LM for distributed training and SGLang for rollout generation within a unified

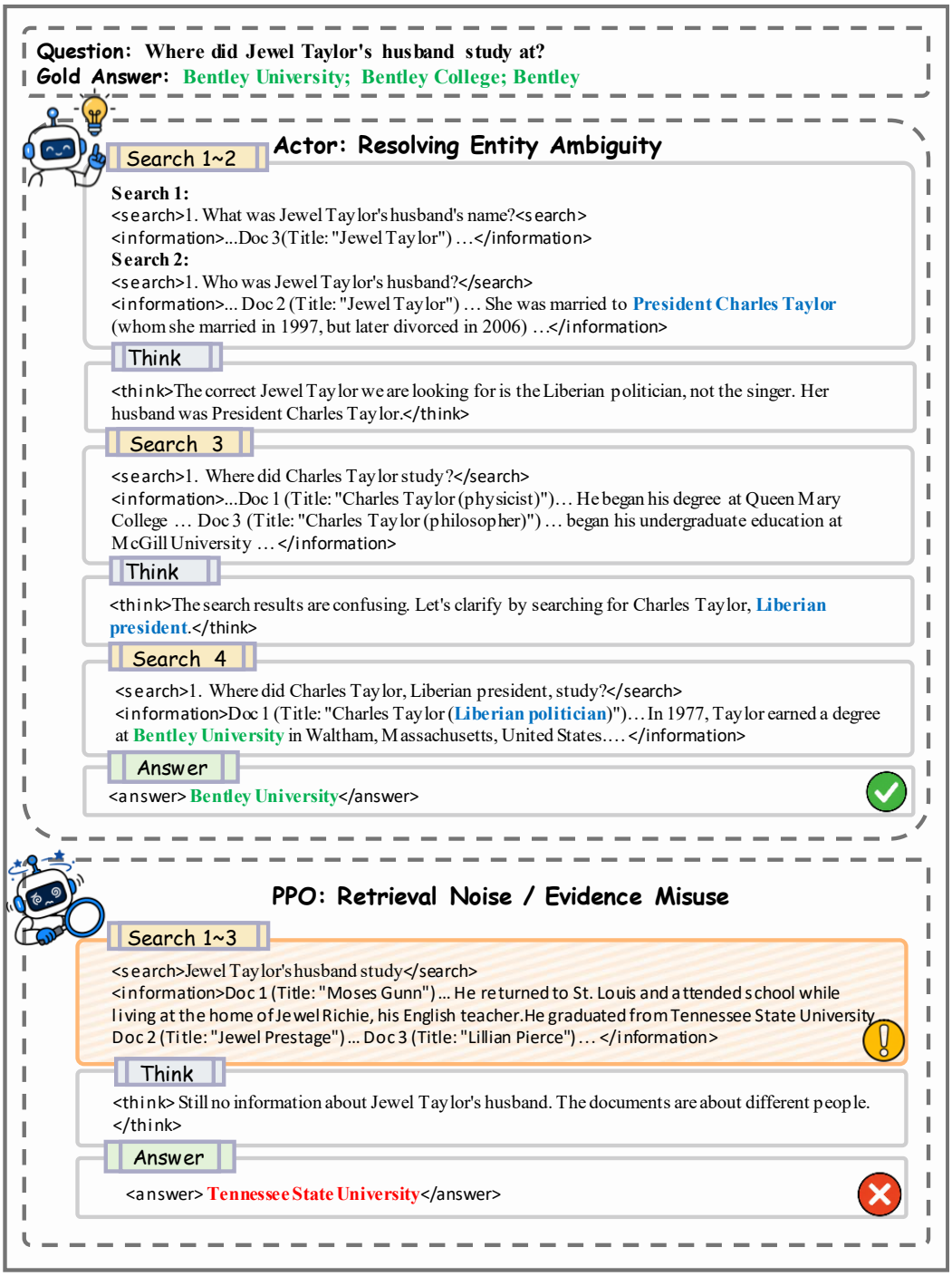

Figure 7: **Retrieval noise and evidence misuse.** PPO allows evidence from unrelated entities to dominate selection and returns Tennessee State University. Ours (actor) resolves Jewel Taylor's husband, disambiguates Charles Taylor as Liberian president, and retrieves Bentley University.

training workflow. The software stack included Python 3.12, PyTorch 2.11, CUDA 12.9, SGLang 0.5.12, and Megatron-Core 0.16 throughout training and evaluation runs. All experiments were conducted on Ubuntu 22.04 servers equipped with NVIDIA A800 GPUs with 80 GB of memory and NVLink interconnects, Intel Xeon Gold 6348 processors, and 1 TB of host memory.



Figure 8: **Under-search.** PPO stops after identifying Sullivan County and returns the intermediate entity as the final answer, leaving the second retrieval step unresolved. Ours (actor) preserves the remaining subgoal, retrieves the county seat, and correctly answers Newport.

F Case Examples

We analyze five aligned evaluation cases under the Search-R1 setting on Qwen2.5-7B-Instruct, where Ours (actor) succeeds and PPO fails despite both trajectories completing normally. Search-R1 requires the agent to formulate queries, integrate retrieved evidence, determine whether further retrieval is needed, and map the resulting evidence to the requested answer. The observed failures fall into five modes. Query decomposition failure leaves a compound information need unresolved across turns. Retrieval noise and evidence misuse allow distractor passages to override the target relation. Under-search terminates at an intermediate entity before completing the required reasoning chain, while over-search continues after sufficient evidence has been obtained and introduces unnecessary distractors. Evidence-to-answer disconnect preserves the relevant facts but selects the wrong entity or semantic role in the final answer. These cases localize errors across query planning, evidence integration, termination, and answer extraction.

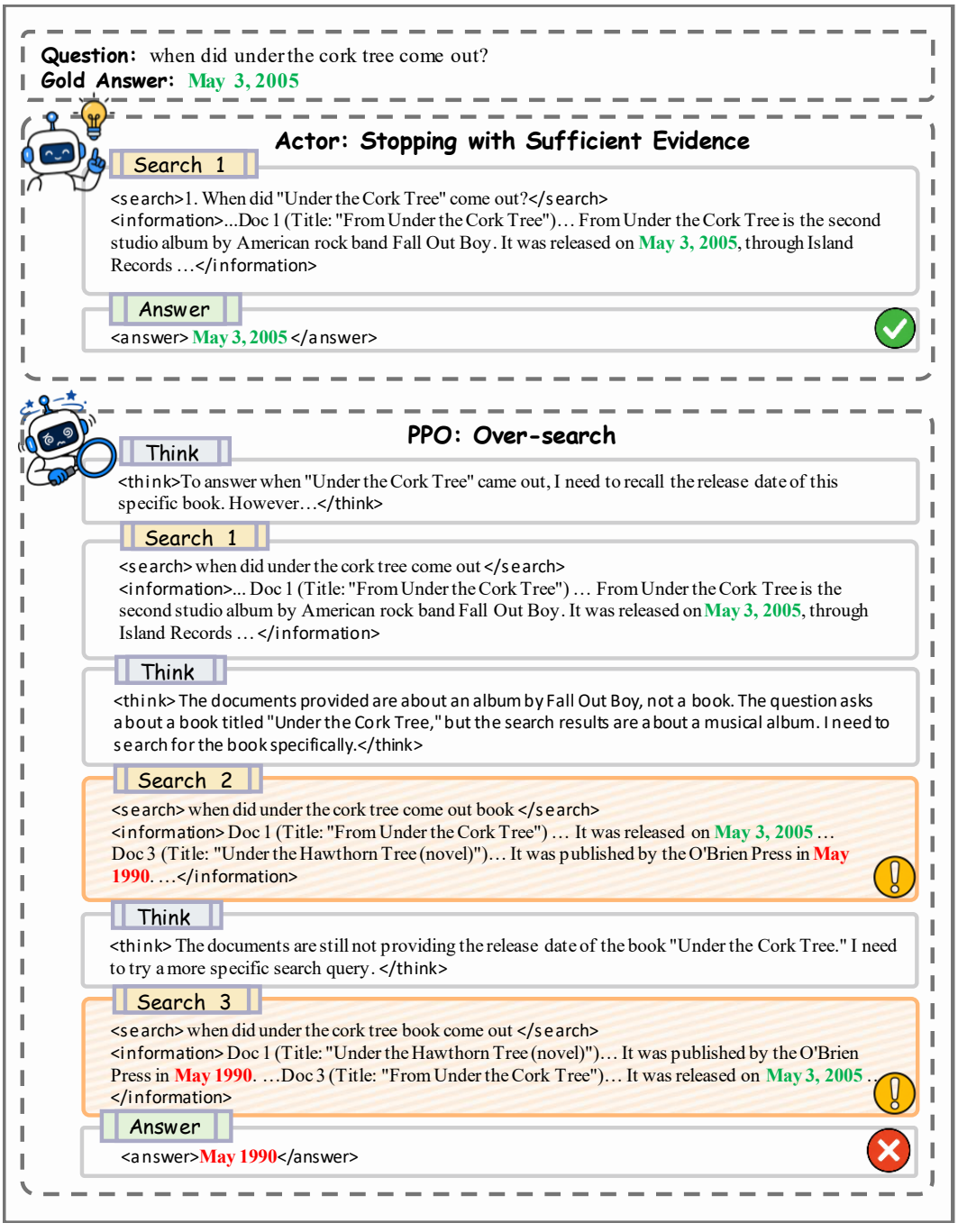

Figure 9: **Over-search.** PPO continues searching after the correct release date is retrieved, retains an unsupported book interpretation, and is diverted to the unrelated May 1990 result. Ours (actor) recognizes sufficient evidence after the first retrieval and stops with May 3, 2005.

Query decomposition failure. PPO repeatedly submits the original compound relation even after retrieval reveals the intermediate entities required to resolve it. The unchanged query state returns nearly identical evidence and exhausts the interaction budget without reaching the answer. This stalled retrieval process is illustrated in Figure 6. Ours (actor) instead converts each resolved intermediate entity into the next retrieval target, progressively reducing the unresolved relation until the final attribute becomes directly searchable. It therefore completes the multi-hop chain with fewer



Figure 10: **Evidence-to-answer disconnect.** PPO retrieves and correctly interprets evidence that André Breton expelled Salvador Dalí, but returns Breton instead of the expelled artist. Ours (actor) preserves the relation direction from evidence to answer and correctly selects Salvador Dalí.

searches. This behavior favors state-changing queries that advance the trajectory over repeated retrieval actions that leave the information state unchanged.

Retrieval noise and evidence misuse. PPO repeats the same ambiguous query, retrieves passages about unrelated individuals, recognizes the mismatch, but ultimately copies an attribute from a distractor. Ours (actor) first resolves the latent entity relation, then augments the query with a discriminative identity attribute before retrieving the requested fact. The corresponding retrieval trajectories are presented in Figure 7 for direct comparison. Its subsequent searches progressively disambiguate the target entity across successive retrieval turns and recover the requested fact. This tighter coupling between evidence assessment and action selection helps reject lexically plausible passages whose entities do not satisfy the relation specified by the question.

Under-search. PPO correctly identifies the two-hop structure but terminates after retrieving only the first-hop entity, returning the intermediate result as the final answer. Ours (actor) preserves the remaining subgoal after the first retrieval, recognizes that the current entity does not match the

requested answer type, and performs the missing lookup. The trajectories are shown in Figure 8, highlighting the divergence in retrieval completion and reasoning. The resulting trajectory maintains task progress across interaction boundaries and conditions termination on answer sufficiency. This behavior preserves the unresolved dependency across turns through each subsequent retrieval and reasoning step and continues retrieval until the full reasoning chain is completed.

Over-search. PPO begins with an unsupported book assumption and keeps searching even after its first retrieval provides the direct answer. Subsequent results contain a lexically similar but unrelated entity whose attribute displaces the initially correct evidence. Ours (actor) recognizes that the first result already resolves the requested relation and terminates immediately. Figure 9 traces how the two policies diverge after sufficient evidence becomes available. Terminating once the evidence is sufficient reduces unnecessary tool use and limits semantic drift from subsequent observations.

Evidence-to-answer disconnect. PPO retrieves a passage that explicitly states the correct relation and reproduces it in the final reasoning, but returns the agent of the event rather than the entity requested by the question. Ours (actor) preserves the direction of the relation when mapping evidence to the answer slot and selects the correct argument. The final evidence-to-answer transition is detailed in Figure 10, highlighting contrasting answer mappings. The difference arises after retrieval, since the required evidence is already available. It reflects more reliable propagation of relational structure from observation to final action through the final prediction stage, extending turn-level credit from evidence acquisition to the answer decision that determines task reward.

G Prompt Templates

Initial Task Prompt

Answer the given question. You must conduct reasoning inside `<think>` and `</think>` first every time you get new information. After reasoning, if you find you lack some knowledge, you can call a search engine by `<search> query </search>` and it will return the top searched results between `<information>` and `</information>`. You can search as many times as you want. If you find no further external knowledge needed, you can directly provide the answer inside `<answer>` and `</answer>`, without detailed illustrations. For example, `<answer> Beijing </answer>`. Question: {question}

Search Result Feedback Template

```
<information>Doc 1(Title: {title_1}) {passage_1}
Doc 2(Title: {title_2}) {passage_2}
Doc 3(Title: {title_3}) {passage_3}</information>
```

Invalid Action Feedback Template

My previous action is invalid. If I want to search, I should put the query between `<search>` and `</search>`. If I want to give the final answer, I should put the answer between `<answer>` and `</answer>`. Let me try again.

The three templates define the interaction protocol used throughout training and evaluation. The initial task prompt specifies the sequence of reasoning, search, and final answer generation. After each valid search action, the environment returns retrieved documents through the search result feedback template. Invalid search or answer formats trigger the invalid action feedback template, which is appended to the interaction history and prompts the model to generate a valid action.