CROSSPO: CROSS-TRAJECTORY POLICY OPTIMIZATION FOR USER-CENTRIC AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

User-centric LLM agents often need to acquire missing information through interaction before they can complete a task. Learning this behavior is challenging because task outcomes reflect both information acquisition and subsequent execution, leaving trajectory-level rewards too coarse to assess individual information requests. We therefore introduce CrossPO, which learns effective information-seeking behavior by aligning equivalent information requests across trajectories. Within each task's rollout batch, it groups approximately equivalent requests and compares average subsequent returns between trajectories containing and omitting each request class. It shrinks the resulting credit toward zero when either group has limited sample support and uses it to augment GRPO advantages while retaining trajectory-level supervision. This allows equivalent requests to share supervision across dialogue histories, so that a request can receive positive local credit even in an unsuccessful trajectory. Experiments on UserGym, ColBench, and τ^2-Bench demonstrate strong overall performance across intent clarification, collaborative coding, and tool use. On ColBench, CrossPO improves pass and success rates over the strongest baseline by up to 5.5 and 6.5 percentage points, respectively. Ablations support the effectiveness of credit assignment and calibration, while behavioral analyses show substantially fewer duplicate requests. Further efficiency analysis shows that CrossPO adds less than 0.5% to per-step training time, making it practical for multi-turn agent training.

1 INTRODUCTION

Recent advances in large language models (LLMs) have enabled interactive agents that collaborate with users to accomplish complex tasks (Yao et al., 2022; Chen et al., 2025; Wu et al., 2025; Shao et al., 2026). For user-centric LLM agents, this collaboration requires understanding user goals, preferences, and constraints that are often underspecified in the initial request (Qian et al., 2024; 2025a). For example, a request such as "plan a trip to Barcelona" may leave the travel dates and budget unspecified, making it difficult to develop a travel plan that meets the user's needs. Agents must resolve such ambiguities through interaction and use the acquired information to guide task execution. Therefore, learning which information to seek and how to use it for task completion is a central challenge for user-centric agents (Qian et al., 2025b; Sun et al., 2025; Kong et al., 2026).

The difficulty is that the usefulness of an information request and the quality of subsequent execution both affect the outcome. An agent may obtain useful information and still fail to complete the task, while a successful interaction may contain unnecessary information requests (Wu et al., 2025; Wang et al., 2026). Reinforcement learning with outcome rewards optimizes task performance, but assigning the same advantage to all generated tokens within a trajectory provides coarse supervision for individual information requests (Shao et al., 2024; Qian et al., 2025b; Wang et al., 2025). To provide finer supervision, several methods introduce turn-level signals that assess the value of individual interactions through learned critics or information-gain objectives (Zhou et al., 2025; Wang et al., 2026; Kong et al., 2026). While effective, these signals rely on critic predictions or changes in policy probabilities. This raises a natural question for user-centric agents: *can credit for individual information requests be derived directly from task outcomes observed across trajectories?*

We therefore introduce CrossPO, which learns effective information-seeking behavior from task outcomes by aligning equivalent information requests across trajectories. Our key observation is that

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.

trajectories for the same task can contain semantically equivalent information requests despite differences in wording, timing, and dialogue history (Gan et al., 2026). These requests provide a common unit for credit assignment: for each request class, we compare subsequent returns between trajectories containing the request and those omitting it. A request can receive positive local credit even in an unsuccessful trajectory when trajectories containing equivalent requests have higher average subsequent returns than those omitting them. This allows equivalent requests to share supervision across dialogue histories, connecting information-seeking decisions to task performance.

Within each task's rollout batch, CrossPO groups approximately equivalent information requests into *information-action anchors*, using native request identifiers or semantic similarity from a frozen embedding model. For each anchor, it estimates local credit from differences in average subsequent returns between trajectories containing and omitting the request. The credit is shrunk toward zero when either group has few samples. The resulting credit augments the GRPO advantage on the model-generated turn that first generates the request, while the original trajectory advantage is retained throughout the interaction. This combines targeted supervision for information acquisition with task supervision for subsequent execution. Credit estimation reuses the sampled trajectories and rewards, without collecting additional rollouts or training a separate critic.

We evaluate CrossPO on UserGym (Qian et al., 2025b), ColBench (Zhou et al., 2025), and τ^2-Bench (Barres et al., 2025), spanning intent clarification, collaborative coding, and tool use. CrossPO achieves strong overall performance across two model backbones, improving ColBench pass and success rates over the strongest baseline by up to 5.5 and 6.5 percentage points, respectively. Our analyses further examine whether performance depends on assigning credit to the corresponding information requests and show that CrossPO substantially reduces duplicate requests. Efficiency analyses show that the additional embedding and clustering for credit estimation account for less than 0.5% of per-step training time, making CrossPO practical for multi-turn agent training.

To summarize, our contributions are:

- We introduce CrossPO, which learns information-seeking behavior in user-centric agents from task outcomes by aligning approximately equivalent requests across interaction histories.
- We develop a request-level credit estimator that compares subsequent returns between trajectories containing and omitting each request, with shrinkage based on the sample support of both groups. The credit augments GRPO without additional rollouts or critic training.
- We demonstrate strong performance on UserGym, ColBench, and τ^2-Bench. Further analyses show substantially fewer duplicate requests and minimal computational overhead.

2 Related Work

User-centric agents. User-centric agents must clarify and satisfy user requirements that are partially specified in the initial request. Research in this area has progressed from understanding user intentions (Qian et al., 2024; Luo et al., 2026) and resolving ambiguity (Rao & Daumé III, 2018; Zhang et al., 2024; Chen et al., 2025; Zhang et al., 2025; Deng et al., 2026) to proactive collaboration (Wu et al., 2025; Acikgoz et al., 2026) and adaptation to user preferences over extended interactions (Sun et al., 2025). Evaluation has expanded to interactions guided by user preferences (Qian et al., 2025a), collaborative content creation (Lee et al., 2022; Zhou et al., 2025; Shao et al., 2026), joint action in shared environments (Barres et al., 2025), and adaptive planning (Liu et al., 2026). Across these settings, collaboration requires agents to connect information acquisition with subsequent task execution, making the selection of useful information requests a central learning problem.

Agentic reinforcement learning. Recent work uses reinforcement learning to train agents for environment interaction (Wang et al., 2025), iterative search (Jin et al., 2025), and user interaction with tool use (Qian et al., 2025b; Zhao et al., 2025). A central challenge is assigning credit to intermediate decisions under trajectory-level supervision (Shao et al., 2024). To provide finer supervision, prior work explores learned critics (Zhou et al., 2024; 2025), signals based on answer or action likelihoods (Wang et al., 2026; Kong et al., 2026; Tao et al., 2026), and counterfactual branches (Li et al., 2026). Such signals can also be constructed by aggregating outcome evidence across sampled trajectories (Feng et al., 2026; Yang et al., 2026; Pang et al., 2026). CrossPO targets a central challenge in user-centric agents: learning which information to seek for task completion. By sharing outcome evidence across semantically equivalent requests, it turns diverse dialogue trajectories

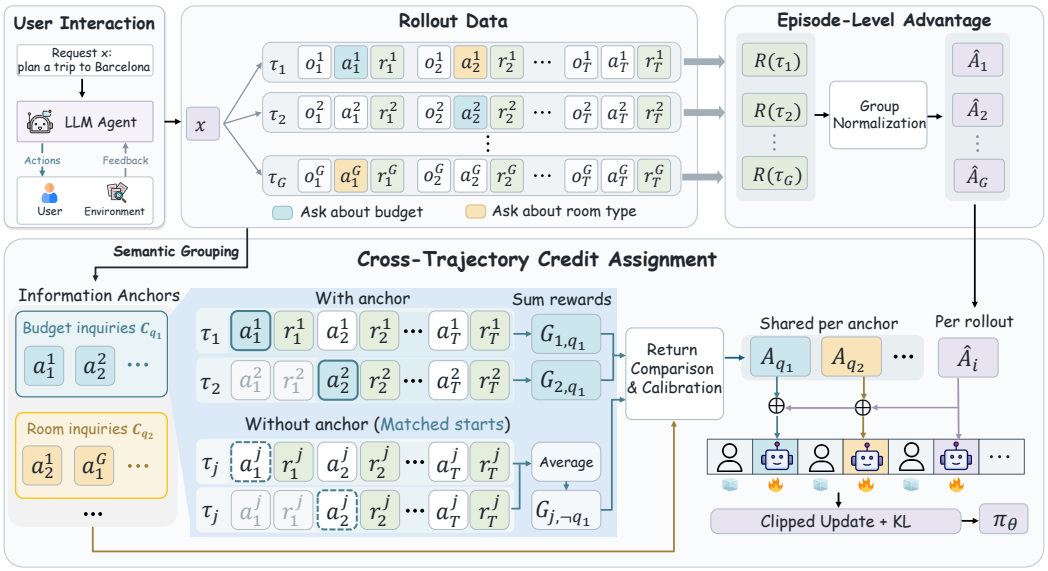


Figure 1: **Overview of CrossPO.** CrossPO groups equivalent information requests across rollouts of the same task and estimates their credit from subsequent returns. The resulting credit augments GRPO advantages to improve information seeking and task execution.

into a common source of calibrated supervision for information acquisition, which links information seeking to task outcomes while retaining trajectory-level supervision.

3 METHOD

We introduce CrossPO, a policy optimization method for user-centric agents that assigns credit to information-seeking actions by comparing subsequent returns across trajectories of the same task. We first formulate the user-centric interaction setting and review GRPO in Section 3.1, then develop the cross-trajectory credit estimator in Section 3.2 and integrate the resulting credit into policy optimization in Section 3.3. The overall training pipeline is illustrated in Figure 1.

3.1 PRELIMINARIES

Problem formulation. We formulate user-centric agent interaction as a finite-horizon partially observable Markov decision process (POMDP) (Kaelbling et al., 1998). Each task $x \sim \mathcal{D}$ begins with a user request that may leave relevant goals, preferences, or constraints unspecified. At step t, the agent receives an observation o_t, such as user feedback or a tool response, and generates an action $a_t \sim \pi_\theta(\cdot \mid h_t, o_t)$, where h_t contains the initial request and preceding actions and observations. The latent state s_t comprises the user's underlying requirements and the environment state. After executing a_t, the environment produces a reward r_t, transitions to $s_{t+1} \sim P(\cdot \mid s_t, a_t)$, and returns $o_{t+1} \sim \Omega(\cdot \mid s_{t+1}, a_t)$. The user or simulator forms part of the environment, with its response process fixed during training. An interaction episode under policy π_θ yields a trajectory $\tau = (o_1, a_1, r_1 \ldots, o_T, a_T, r_T)$ with a task-defined episode score $R(\tau)$. Our objective is

$$\mathcal{J}_{\text{task}}(\theta) = \mathbb{E}_{x\sim\mathcal{D},\, \tau\sim\pi_\theta(\cdot|x)}\big[R(\tau)\big]. \tag{1}$$

Information-seeking actions, including user questions, search queries, and information-retrieval tool calls, can clarify user requirements or provide evidence for subsequent decisions.

Group relative policy optimization. We adopt Group Relative Policy Optimization (GRPO) (Shao et al., 2024) as the underlying policy optimization base. GRPO estimates relative advantages from the rewards of sampled rollouts without training a separate critic. For each task $x \sim \mathcal{D}$, the old policy $\pi_{\theta_{\text{old}}}$ samples a group of G outputs $\{o_i\}_{i=1}^{G}$, each receiving a reward r_i. The policy is optimized by

maximizing the clipped surrogate objective with KL regularization:

$$\mathcal{J}_{\mathrm{GRPO}}(\theta) = \mathbb{E}_{x\sim\mathcal{D},\,\{o_i\}\sim\pi_{\theta_{\mathrm{old}}}(\cdot|x)}\left[\frac{1}{G}\sum_{i=1}^{G}\frac{1}{|o_i|}\sum_{t=1}^{|o_i|}\left\{\min\left(\frac{\pi_\theta(o_{i,t}\mid x,o_{i,<t})}{\pi_{\theta_{\mathrm{old}}}(o_{i,t}\mid x,o_{i,<t})}\widehat{A}_i,\right.\right.\right.$$
$$\left.\left.\left.\mathrm{clip}\left(\frac{\pi_\theta(o_{i,t}\mid x,o_{i,<t})}{\pi_{\theta_{\mathrm{old}}}(o_{i,t}\mid x,o_{i,<t})},1-\epsilon,1+\epsilon\right)\widehat{A}_i\right)-\lambda_{\mathrm{KL}}\mathbb{D}_{\mathrm{KL}}(\pi_\theta\|\pi_{\mathrm{ref}})\right\}\right]. \tag{2}$$

Here, $\widehat{A}_i = \dfrac{r_i - \mathrm{mean}(r_1,\ldots,r_G)}{\mathrm{std}(r_1,\ldots,r_G)}$ is the group-relative advantage, shared by all tokens in o_i. The parameter ϵ controls clipping, and λ_{KL} weights the KL penalty toward the reference policy π_{ref}, evaluated at the corresponding token context. This shared advantage provides trajectory-level supervision but does not distinguish the contributions of individual information-seeking actions.

3.2 Cross-Trajectory Credit Estimation

CrossPO estimates credit in three stages: grouping approximately equivalent information requests into anchors, comparing subsequent returns between trajectories with and without each request, and calibrating the resulting contrasts for sample support and return scale. All three stages operate within each task's sampled rollout group.

Information-action anchors. Across rollouts of the same task, information-seeking actions may target the same information despite differences in wording or interaction history. We introduce *information-action anchors* to align these requests for subsequent return comparisons. For task x, let $\mathcal{E} \subseteq \{1,\ldots,G\}$ index eligible trajectories after excluding infrastructure failures. We group their information-seeking actions by the information requested and denote the anchor set by $\mathcal{Q}_x$. Each anchor $q \in \mathcal{Q}_x$ identifies a class C_q of approximately equivalent requests (e.g., budget inquiries).

To construct anchors across differently worded requests, we cluster compatible free-form requests using unit-normalized representations $\mathbf{e}(a)$ from a frozen embedding model; native request identifiers are used directly when available. We use complete linkage so that every pair of requests within a cluster meets a similarity threshold η. For two clusters C and C', their similarity is $\mathrm{sim}(C,C') = \min\{\mathbf{e}(a)^\top\mathbf{e}(a') \mid a \in C,\, a' \in C'\}$. Starting from singleton clusters, we repeatedly merge the most similar pair with similarity at least η, stopping when no such pair remains.

For each anchor $q \in \mathcal{Q}_x$, we partition eligible trajectories into those containing at least one request in C_q and those containing none:

$$\mathcal{I}_q = \{i \in \mathcal{E} : \exists\, t,\ a_{i,t} \in C_q\}, \qquad \mathcal{I}_{\neg q} = \{i \in \mathcal{E} : \forall\, t,\ a_{i,t} \notin C_q\}. \tag{3}$$

Let $n_q = |\mathcal{I}_q|$ and $n_{\neg q} = |\mathcal{I}_{\neg q}|$ denote the respective group sizes. Each trajectory is counted once, even if it contains repeated requests. If either group is empty, no return comparison is possible, and we assign zero credit to the anchor.

Subsequent return comparison. To assess an information request through subsequent outcomes, we exclude rewards obtained before the request. For each trajectory $i \in \mathcal{I}_q$, the first occurrence of anchor q is at $t_{i,q} = \min\{t : a_{i,t} \in C_q\}$, from which we compute the subsequent return:

$$G_{i,q} = \sum_{u=t_{i,q}}^{T_i} r_{i,u}, \tag{4}$$

where T_i is the number of interaction steps in trajectory i. Rewards are summed without discounting so that delayed task outcomes retain their full weight. Using the first occurrence gives each trajectory one return sample, preventing repeated requests from increasing its weight in the comparison.

Trajectories omitting q have no corresponding request time. To avoid comparing their full-episode rewards with the requesting group's subsequent returns, we use the observed request times as reference steps. For each trajectory $j \in \mathcal{I}_{\neg q}$, we compute its return separately from every first-occurrence

time $t_{i,q}$, $i \in \mathcal{I}_q$, and average these n_q returns to obtain a single reference value:

$$G_{j,\neg q} = \frac{1}{n_q} \sum_{i \in \mathcal{I}_q} \begin{cases} \sum_{u=t_{i,q}}^{T_j} r_{j,u}, & t_{i,q} \le T_j, \\ r_{j,T_j}, & t_{i,q} > T_j, \text{ normal termination}, \\ 0, & t_{i,q} > T_j, \text{ truncation}. \end{cases} \tag{5}$$

If a trajectory ends normally before the reference step, we retain its final reward to preserve the episode's outcome, since summing rewards from that step onward would yield zero. If it is truncated before that step, we use zero because no subsequent rewards are observed. Each reference trajectory contributes one averaged return, and the difference in group means gives

$$\widehat{\Delta}_q = \frac{1}{n_q} \sum_{i \in \mathcal{I}_q} G_{i,q} - \frac{1}{n_{\neg q}} \sum_{j \in \mathcal{I}_{\neg q}} G_{j,\neg q}. \tag{6}$$

A positive $\widehat{\Delta}_q$ indicates higher subsequent returns for trajectories requesting q, providing an outcome-based signal for assigning credit to the request. With terminal-only rewards and normal termination, this contrast reduces to the difference in mean terminal rewards. We examine the relationship between the resulting anchor credit and task success in Section 5.1.

Anchor credit calibration. The trajectories with an anchor and those without it may be highly imbalanced in number, allowing an unusually high or low return in the smaller group to produce a large contrast in Eq. (6). To reduce the influence of such comparisons, we introduce a shrinkage factor ρ_q that scales the contrast before assigning anchor credit:

$$\rho_q = \frac{n_{\text{eff},q}}{n_{\text{eff},q} + \kappa}, \qquad n_{\text{eff},q} = \frac{n_q n_{\neg q}}{n_q + n_{\neg q}}, \tag{7}$$

where $\kappa \ge 0$ controls the strength of shrinkage. For $\kappa > 0$ and a fixed trajectory count, greater imbalance between the groups gives a smaller ρ_q. This reduces the magnitude of the contrast without changing its sign, so that comparisons with few trajectories on either side contribute less to the policy update. The choice of $n_{\text{eff},q}$ is motivated by a variance calculation: for independent samples with common variance σ^2, the difference in group means has variance $\frac{\sigma^2}{n_{\text{eff},q}}$. This inverse relationship motivates stronger shrinkage when $n_{\text{eff},q}$ is small. We provide the derivation in Appendix B.3.

To account for differences in return scale, let s_q denote the sample standard deviation of the combined comparison returns, with one entry per trajectory:

$$s_q = \text{Std}\big((G_{i,q})_{i \in \mathcal{I}_q}, (G_{j,\neg q})_{j \in \mathcal{I}_{\neg q}}\big). \tag{8}$$

Combining normalization with the shrinkage factor ρ_q, we define the *anchor-level credit* as

$$A_q = \text{clip}\left(\rho_q \frac{\widehat{\Delta}_q}{\max(s_q, \sigma_{\min})}, -a_{\max}, a_{\max}\right), \tag{9}$$

where $\sigma_{\min} > 0$ lower-bounds the normalization scale and $a_{\max} > 0$ bounds the credit magnitude. Normalization expresses the contrast relative to the observed return variation, while the scale floor prevents a very small standard deviation from amplifying minor absolute differences. Clipping further limits the magnitude of the auxiliary signal. The resulting credit supplies a local correction to the GRPO advantage through the action-level assignment below.

3.3 Credit-Augmented Policy Optimization

Anchor credit summarizes the outcome evidence for equivalent information requests across rollouts. We use this evidence to refine the advantages of the corresponding actions while retaining GRPO's trajectory-level supervision.

Action-level credit assignment. To match the return estimator based on one sample per trajectory, we assign each anchor's credit only to its first occurrence within a rollout. For an eligible rollout $i \in \mathcal{E}$, a token $o_{i,k}$ in the recorded span of a single information-seeking action $a_{i,t} \in C_q$ receives the augmented advantage

$$\widehat{A}_{i,k}^{\text{CrossPO}} = \widehat{A}_i + \beta \left(\mathbf{1}\{t = t_{i,q}\} A_q - c_0\right), \tag{10}$$

where $\widehat{A}_i$ is the GRPO advantage, $\beta \geq 0$ controls the contribution of local credit, and $c_0 \geq 0$ is an interaction cost in advantage units. The first occurrence receives the correction $\beta(A_q - c_0)$, whereas later occurrences receive only $-\beta c_0$. Repeated requests therefore incur a cost without accumulating the same estimated benefit.

The correction is applied to all model-generated tokens in the assistant turn that produces the information-seeking action. Each token receives the full correction in Eq. (10), rather than a share divided among the turn's tokens. All other model-generated tokens retain the GRPO advantage $\widehat{A}_i$.

Policy update. We incorporate the local credit into policy optimization by replacing the shared GRPO advantage with $\widehat{A}^{\mathrm{CrossPO}}_{i,k}$. For each task $x \sim \mathcal{D}$, we sample G rollouts under $\pi_{\theta_{\mathrm{old}}}$ and write the clipped surrogate objective with KL regularization as

$$\mathcal{J}_{\mathrm{CrossPO}}(\theta) = \mathbb{E}_{x\sim\mathcal{D},\,\{o_i\}\sim\pi_{\theta_{\mathrm{old}}}(\cdot|x)}\left[\frac{1}{\sum_{j=1}^{G}|o_j|}\sum_{i=1}^{G}\sum_{k=1}^{|o_i|}\left\{\min\left(\frac{\pi_\theta(o_{i,k}\mid x,o_{i,<k})}{\pi_{\theta_{\mathrm{old}}}(o_{i,k}\mid x,o_{i,<k})}\widehat{A}^{\mathrm{CrossPO}}_{i,k},\right.\right.\right.$$
$$\left.\left.\left.\mathrm{clip}\left(\frac{\pi_\theta(o_{i,k}\mid x,o_{i,<k})}{\pi_{\theta_{\mathrm{old}}}(o_{i,k}\mid x,o_{i,<k})},1-\epsilon,1+\epsilon\right)\widehat{A}^{\mathrm{CrossPO}}_{i,k}\right) - \lambda_{\mathrm{KL}}\mathbb{D}_{\mathrm{KL}}(\pi_\theta\|\pi_{\mathrm{ref}})\right\}\right]. \tag{11}$$

Here, $|o_i|$ counts model-generated tokens included in the loss, and $o_{i,<k}$ includes preceding agent outputs and user or tool responses. User and tool responses remain in the context but are excluded from the loss, along with padding. Training averages the loss over these model-generated tokens across sampled groups, with the KL penalty evaluated at the corresponding token context. We retain the baseline's dual clipping for negative advantages. All advantages and anchor assignments remain fixed during optimization. The auxiliary credit modifies advantages without changing environment rewards, and setting $\beta = 0$ restores the GRPO advantages. Credit estimation reuses the sampled rollouts, requiring no additional environment interactions or policy evaluations.

4 Experiments

4.1 Experiment Setup

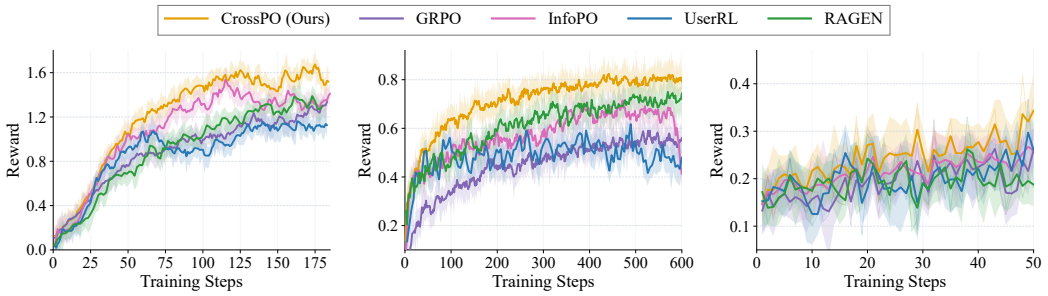
Benchmarks and Metrics. We evaluate CrossPO on three multi-turn benchmarks covering intent clarification, collaborative coding, and tool use. **UserGym** (Qian et al., 2025b) contains eight environments with a unified Action-Search-Answer interface. We train on *Function*, *Persuade*, *Tau*, *Travel*, and *Turtle*, and reserve *Intention*, *Telepathy*, and *Search* to evaluate cross-task generalization. Following the official evaluation protocol, we report the official score for each environment. **ColBench** (Zhou et al., 2025) is a collaborative programming benchmark in which an agent interacts with a simulated user to clarify missing implementation details and produce functionally correct code. We evaluate on the complete backend programming test set and report Pass, the average fraction of tests passed, and Succ, the proportion of solutions passing all tests. **τ^2-Bench** (Barres et al., 2025) evaluates dual-control interaction in airline, retail, and telecommunications domains, where both the agent and user may modify a shared environment. We evaluate each task over four independent trials and report $\mathrm{Avg@4}$, the mean task reward across these trials.

Baselines. We compare CrossPO with four RL baselines. GRPO (Shao et al., 2024) serves as the standard outcome-reward baseline. UserRL (Qian et al., 2025b) uses length-normalized trajectory scoring and future-reward credit assignment, while RAGEN (Wang et al., 2025) improves multi-turn training stability through trajectory filtering and decoupled clipping. InfoPO (Kong et al., 2026) assigns turn-level credit using counterfactual information gain.

Implementation Details. We conduct all experiments on Qwen2.5-7B-Instruct and Qwen3-4B and use GPT-4o-mini as the shared user simulator across all methods. Across all methods, we use a learning rate of 1×10^{-6}, sample 8 rollouts per prompt, and set the global batch size to 64 for UserGym and 32 for both ColBench and τ^2-Bench. For CrossPO, we encode free-form information-seeking actions with Qwen3-Embedding-0.6B and use a similarity threshold of 0.70. We use $\kappa = 1$, $\sigma_{\min} = 0.1$, and $\beta = 0.5$, with A_q clipped to $[-1.25, 1.25]$. For fair and controlled comparisons, all methods use identical training budgets, rollout configurations, user-simulator settings, and evaluation protocols. Further details on the training hyperparameter settings are provided in Appendix C.

Table 1: Performance comparison on UserGym, ColBench, and τ^2-Bench. Within the closed-source model group and each Qwen backbone group, the best results are shown in **bold** and the second-best results are underlined. Avg denotes the unweighted mean of the 13 reported metrics.

Method	UserGym								ColBench		τ^2-Bench			Avg
	Travel	Func.	Pers.	Tau	Turtle	*Search*	*Inten.*	*Tele.*	Pass	Succ.	Telec.	Retail	Air	
Closed-Source Models														
Gemini-3-Flash	0.574	**0.423**	**0.695**	**0.167**	0.153	**0.968**	1.718	**0.829**	0.515	0.382	**0.469**	**0.619**	**0.558**	**0.621**
GPT-4.1	0.554	0.051	0.599	0.109	**0.267**	0.480	**1.867**	0.732	**0.529**	**0.403**	0.388	0.544	0.400	0.533
GPT-4o-mini	**0.596**	0.089	0.683	0.091	0.117	0.568	1.277	0.512	0.463	0.342	0.113	0.400	0.175	0.417
Qwen2.5-7B-Instruct														
Base	0.441	0.026	0.290	0.012	0.062	0.376	1.253	0.293	0.247	0.146	0.144	0.131	0.075	0.269
GRPO	0.526	0.103	0.401	0.030	0.158	0.416	1.805	0.415	0.421	0.316	0.156	0.163	0.088	0.384
UserRL	0.546	0.115	0.444	0.048	0.152	0.432	1.825	0.439	0.436	0.327	0.169	0.169	0.075	0.398
RAGEN	0.538	0.128	0.548	0.012	0.148	0.448	1.815	0.463	0.449	0.348	0.175	0.175	0.150	0.415
InfoPO	0.588	0.167	0.536	**0.091**	0.178	0.480	1.892	0.488	0.534	0.426	0.181	0.188	**0.163**	0.455
CrossPO (Ours)	**0.592**	**0.179**	**0.563**	0.085	**0.206**	**0.560**	**1.925**	**0.512**	**0.589**	**0.491**	**0.256**	**0.200**	0.150	**0.485**
Qwen3-4B														
Base	0.482	0.064	0.433	0.055	0.104	0.464	1.440	0.463	0.256	0.146	0.081	0.206	0.062	0.327
GRPO	0.524	0.090	0.468	0.061	0.126	0.632	1.495	0.512	0.423	0.326	0.094	0.219	0.075	0.388
UserRL	0.538	0.090	0.508	0.073	0.121	0.696	1.732	0.537	0.468	0.342	0.100	0.163	0.075	0.419
RAGEN	0.528	0.103	0.512	0.018	0.117	0.712	1.572	0.488	0.479	0.361	0.137	0.225	0.088	0.411
InfoPO	0.589	0.115	**0.556**	**0.097**	0.154	0.848	1.862	0.542	0.553	0.439	0.156	0.244	0.100	0.481
CrossPO (Ours)	**0.596**	**0.167**	**0.556**	0.091	**0.199**	**0.856**	**1.895**	**0.561**	**0.576**	**0.485**	**0.238**	**0.325**	**0.175**	**0.517**



(a) UserGym (b) ColBench (c) τ²-Bench

Figure 2: **Training dynamics across the three benchmarks.** We compare the training rewards of CrossPO and representative RL baselines using Qwen3-4B as the actor backbone on (a) UserGym, (b) ColBench, and (c) τ^2-Bench. Solid lines show the mean reward across three random seeds, while shaded regions represent mean $\pm$ standard deviation.

4.2 Main Results

Overall performance. Table 1 shows that CrossPO achieves strong performance across all three benchmarks on Qwen2.5-7B-Instruct and Qwen3-4B, with average scores of 0.485 and 0.517, respectively. On UserGym, it outperforms GRPO in all eight environments on both backbones, with further improvements over InfoPO on *Function* and *Turtle*. For example, on Qwen3-4B, these two scores increase from 0.115 to 0.167 and from 0.154 to 0.199, respectively. The gains extend to collaborative coding on ColBench, where CrossPO improves both *Pass* and *Succ.* over InfoPO on both backbones. In particular, the proportion of solutions passing all tests increases by 6.5 and 4.6 percentage points on Qwen2.5-7B-Instruct and Qwen3-4B, respectively. For tool use on τ^2-Bench, CrossPO surpasses InfoPO on *Telecom* and *Retail* with both backbones. On Qwen3-4B, the improvements cover all three domains, including an increase in *Retail* from 0.244 to 0.325. Qualitative examples of agent interactions are provided in Appendix D.

Generalization to held-out environments. *Search*, *Intention*, and *Telepathy* are held out from UserGym training to evaluate generalization. On both backbones, CrossPO outperforms all compared RL baselines in each of these three environments. On Qwen3-4B, its *Telepathy* score reaches 0.561, compared with 0.542 for InfoPO. On Qwen2.5-7B-Instruct, the *Search* score increases from 0.480 to 0.560 relative to InfoPO, while *Telepathy* improves from 0.488 to 0.512. These results sup-

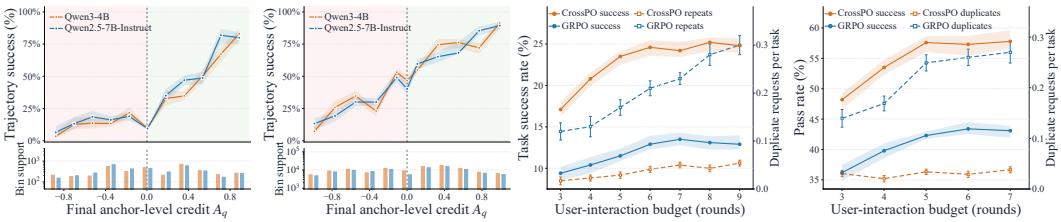


Figure 3: (a) Credit-binned trajectory success rates on τ^2-Bench. (b) Credit-binned trajectory success rates on ColBench. (c) Averaged task success on τ^2-Bench under varying user-interaction budgets. (d) Pass rate on ColBench under varying user-interaction budgets. In (c,d), solid lines show performance (left axes), while dashed lines show duplicate requests per task (right axes).

port the generalization ability of CrossPO, with performance gains extending beyond the training environments to held-out interaction tasks in UserGym.

5 Analysis

5.1 Mechanism Analysis

Training Dynamics. CrossPO achieves faster and more sustained optimization across all three benchmarks. Figure 2 compares the training dynamics of CrossPO and representative RL baselines using Qwen3-4B as the actor backbone. On UserGym and ColBench, CrossPO improves rapidly during the early stages and maintains the highest reward throughout later training. On τ^2-Bench, despite greater reward fluctuations, CrossPO continues to improve and reaches the strongest final performance. This consistent pattern supports the central role of suffix-return credit in CrossPO: it converts sparse terminal feedback into discriminative action-level supervision and directs policy updates toward information requests that benefit subsequent decisions.

CrossPO assigns larger final anchor-level credits A_q to information actions associated with stronger downstream task performance. We partition training samples from τ^2-Bench and ColBench into bins according to their credit A_q. Within each bin, we report the proportion of samples from successful trajectories. As shown in Figure 3(a,b), trajectory success rates exhibits a clear upward trend as A_q increases from negative to positive values. This alignment highlights CrossPO's central advantage of converting sparse terminal rewards into discriminative action-level supervision through suffix-return comparisons between trajectories with and without a given information action. CrossPO therefore distinguishes information-seeking decisions by their downstream utility rather than relying solely on a shared trajectory-level signal for all intermediate actions.

CrossPO uses user interaction more effectively than GRPO, attaining higher pass rates under every budget while making substantially fewer duplicate requests. To compare interaction quality, we evaluate Qwen3-4B models trained with CrossPO and GRPO on τ^2-Bench and ColBench. Within each benchmark, we vary the user-interaction budget and evaluate both methods on the same tasks at each budget level. Figure 3(c,d) reports task performance and the average number of duplicate requests per task on τ^2-Bench and ColBench. CrossPO achieves larger overall performance gains than GRPO as the interaction budget increases on both benchmarks. At the largest evaluated budget in each benchmark, CrossPO also makes **over 80% fewer duplicate requests**, indicating less redundant interaction despite the greater allowance for user communication. These results indicate that CrossPO achieves better outcomes with less redundant interaction under matched budgets. Its suffix-return credit assigns greater weight to questions that elicit necessary information, encouraging targeted information seeking while reducing repeated requests.

Robustness to the User Simulator. We examine the sensitivity of CrossPO to the choice of user simulator by comparing GPT-4o-mini, the default simulator, with GPT-4o on UserGym, ColBench, and τ^2-Bench. All other experimental settings are kept unchanged. Replacing GPT-4o-mini with GPT-4o improves all reported metrics, with the largest absolute gain observed in the ColBench pass rate ($+0.060$).

Table 2: **Robustness to the user simulator.** Performance of CrossPO with Qwen3-4B across three benchmarks under different user simulators.

Method	UserGym	ColBench		τ^2-Bench
	Avg.	Pass	Succ.	Avg.
GPT-4o-mini (Default)	0.615	0.576	0.485	0.246
GPT-4o	0.626	0.636	0.515	0.263

The UserGym average, ColBench success rate, and τ^2-Bench average also increase by 0.011, 0.030, and 0.017, respectively. These results suggest that CrossPO remains effective under different user simulators and that its performance is not specific to GPT-4o-mini.

5.2 Ablation.

Ablation Study. We conduct controlled ablations using Qwen3-4B across UserGym, ColBench, and τ^2-Bench. To isolate the contribution of each component in CrossPO, we individually remove standardization, small-sample shrinkage, and clipping while keeping all other training and evaluation settings unchanged. We also include a shuffled-credit control that permutes the action-level credits across decision points. This control preserves the overall credit distribution while breaking the correspondence between each action and its original credit A_q. Table 3 reports the aggregate scores on UserGym and τ^2-Bench, together with pass rate and success rate on ColBench. Removing any of the three components reduces performance across all reported metrics, with the largest decline caused by removing small-sample shrinkage. Shuffled credit yields the lowest scores across all benchmarks, supporting the effectiveness of the action-level supervision derived from suffix-return comparisons.

Table 3: Component ablations and credit-control results on Qwen3-4B across three benchmarks.

Method	UserGym	ColBench		τ^2-Bench
	Avg.	Pass	Succ.	Avg.
CrossPO	**0.615**	**0.576**	**0.485**	**0.246**
Component Ablations				
w/o Standardization	0.583	0.542	0.437	0.231
w/o Shrinkage	0.569	0.526	0.391	0.185
w/o Clipping	0.608	0.569	0.435	0.229
Credit Control				
Shuffled Credit	0.452	0.377	0.283	0.117

5.3 Training Efficiency.

CrossPO's additional embedding and clustering account for less than 0.5% of the measured per-step runtime. We measure the average per-step runtime of each component when training Qwen3-4B with CrossPO on τ^2-Bench using 4 NVIDIA A800 GPUs. CrossPO and GRPO share rollout generation, old-policy and reference-policy log-probability computation, and actor optimization. CrossPO additionally performs semantic embedding and clustering to support suffix-return comparisons within existing rollout groups, requiring neither additional rollouts nor additional policy rescoring. As shown in Figure 4, rollout generation and actor optimization dominate the computational cost, whereas embedding and clustering require only 3.22 and 0.79 seconds per step, respectively. These two CrossPO-specific operations account for only 0.37% and 0.09% of the total per-step runtime. These results demonstrate that CrossPO enables fine-grained decision credit assignment while incurring negligible additional computational overhead.

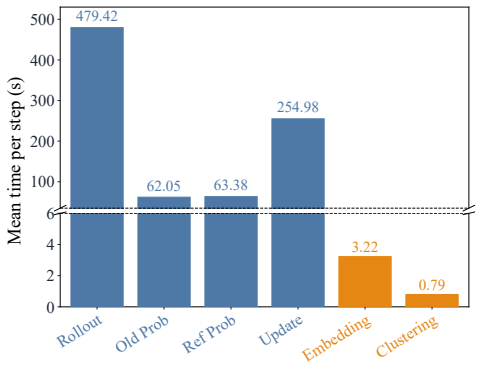


Figure 4: Per-step training time breakdown of CrossPO with Qwen3-4B on τ^2-Bench. Blue bars indicate components shared with GRPO, while orange bars represent CrossPO-specific evidence embedding and clustering. The broken y-axis improves the visibility of low-cost components.

6 Conclusion

We introduced CrossPO, which learns information-seeking behavior in user-centric agents by aligning semantically equivalent information requests across trajectories. It compares subsequent returns between trajectories that include and exclude each request to derive calibrated local credit that augments GRPO. Experiments on UserGym, ColBench, and τ^2-Bench demonstrate strong performance, with analyses showing substantially fewer duplicate requests and minimal training overhead. These findings establish cross-trajectory outcome comparison as a practical approach to learning effective interaction policies without additional rollouts or critic training. Future work will extend the method to longer, more diverse interactions with real users.

AI USE STATEMENT

In this work, we used generative AI tools, including OpenAI ChatGPT, to assist with writing, literature searches, and LaTeX formatting. These tools were not used to develop the research idea, method, or experimental implementation. We reviewed and verified all AI-assisted content and take full responsibility for the final paper and supplementary materials.

ETHICS STATEMENT

We adhere to the ICLR Code of Ethics.

Safety of content. Our experiments are conducted within established benchmark environments using simulated users. We did not observe violent, offensive, or otherwise harmful content in the data and generated outputs used in this study.

Privacy and intellectual property. We use publicly available benchmarks and existing models, without collecting private conversations or personal information from real users. We use these resources in accordance with their respective licenses and applicable terms of use.

REPRODUCIBILITY STATEMENT

We provide the implementation of CrossPO and training code in a ZIP archive submitted as supplementary material. The code supports training across the benchmark environments used in this study, including UserGym, ColBench, and τ^2-Bench. Section 3 and Appendix A detail the algorithm and credit estimator, while Section 4.1 and Appendix C describe the experimental setup, training hyperparameters, and evaluation protocols. Together, these materials support independent reproduction of the training procedure and reported comparisons.

REFERENCES

Emre Can Acikgoz, Jinoh Oh, Jie Hao, Joo Hyuk Jeon, Heng Ji, Dilek Hakkani-Tur, Gokhan Tur, Xiang Li, Chengyuan Ma, and Xing Fan. Speakrl: Synergizing reasoning, speaking, and acting in language models with reinforcement learning. In *Proceedings of the 16th International Workshop on Spoken Dialogue System Technology*, pp. 312–325, 2026.

Victor Barres, Honghua Dong, Soham Ray, Xujie Si, and Karthik Narasimhan. τ^2-bench: Evaluating conversational agents in a dual-control environment. *arXiv preprint arXiv:2506.07982*, 2025.

Maximillian Chen, Ruoxi Sun, Tomas Pfister, and Sercan Arik. Learning to clarify: Multi-turn conversations with action-based contrastive self-training. In *International Conference on Learning Representations*, volume 2025, pp. 32244–32279, 2025.

Mengyi Deng, Zhiwei Li, Xin Li, Tingyu Zhu, Ying Zhao, Zhijiang Guo, and Wei Wang. Uncertainty-aware clarification in llm agents with information gain. *arXiv preprint arXiv:2606.03135*, 2026.

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, 2026.

Yujian Gan, Changling Li, Jinxia Xie, Luou Wen, Matthew Purver, and Massimo Poesio. Clarq4llm: A benchmark for models clarifying and requesting information in task-oriented dialog. *IEEE Transactions on Audio, Speech and Language Processing*, 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.

Leslie Pack Kaelbling, Michael L Littman, and Anthony R Cassandra. Planning and acting in partially observable stochastic domains. *Artificial intelligence*, 101(1-2):99–134, 1998.

Fanqi Kong, Jiayi Zhang, Mingyi Deng, Chenglin Wu, Yuyu Luo, and Bang Liu. Infopo: Information-driven policy optimization for user-centric agents. *arXiv preprint arXiv:2603.00656*, 2026.

Mina Lee, Percy Liang, and Qian Yang. Coauthor: Designing a human-ai collaborative writing dataset for exploring language model capabilities. In *Proceedings of the 2022 CHI conference on human factors in computing systems*, pp. 1–19, 2022.

Mingchen Li, Hansi Zeng, Zhuo Qian, Jiatan Huang, Sunjae Kwon, Hamed Zamani, and Hong Yu. Rice-po: Turning retrieval interactions into credit signals for reasoning agents. *arXiv preprint arXiv:2605.26352*, 2026.

Jiayu Liu, Cheng Qian, Zhenhailong Wang, Bingxuan Li, Jiateng Liu, Qing Zong, Heng Wang, Jeonghwan Kim, Yumeng Wang, Bingxiang He, et al. Adaplanbench: Evaluating adaptive planning in large language model agents under world and user constraints. *arXiv preprint arXiv:2606.05622*, 2026.

Haohao Luo, Zexi Li, Yuexiang Xie, Wenhao Zhang, Yaliang Li, and Ying Shen. Intentrl: Training proactive user-intent agents for open-ended deep research via reinforcement learning. *arXiv preprint arXiv:2602.03468*, 2026.

Colin McDiarmid et al. On the method of bounded differences. *Surveys in combinatorics*, 141(1): 148–188, 1989.

Cong Pang, Xuyu Feng, Yujie Yi, Jiaqi Su, Zixuan Chen, Jiawei Hong, Tiankuo Yao, Nang Yuan, Jiapeng Luo, Lewei Lu, et al. Ica: Information-aware credit assignment for visually grounded long-horizon information-seeking agents. *arXiv preprint arXiv:2602.10863*, 2026.

Cheng Qian, Bingxiang He, Zhong Zhuang, Jia Deng, Yujia Qin, Xin Cong, Zhong Zhang, Jie Zhou, Yankai Lin, Zhiyuan Liu, et al. Tell me more! towards implicit user intention understanding of language model driven agents. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1088–1113, 2024.

Cheng Qian, Zuxin Liu, Akshara Prabhakar, Zhiwei Liu, Jianguo Zhang, Haolin Chen, Heng Ji, Weiran Yao, Shelby Heinecke, Silvio Savarese, et al. Userbench: An interactive gym environment for user-centric agents. *arXiv preprint arXiv:2507.22034*, 2025a.

Cheng Qian, Zuxin Liu, Akshara Prabhakar, Jielin Qiu, Zhiwei Liu, Haolin Chen, Shirley Kokane, Heng Ji, Weiran Yao, Shelby Heinecke, et al. Userrl: Training interactive user-centric agent via reinforcement learning. *arXiv preprint arXiv:2509.19736*, 2025b.

Sudha Rao and Hal Daumé III. Learning to ask good questions: Ranking clarification questions using neural expected value of perfect information. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2737–2746, 2018.

Yijia Shao, Vinay Samuel, Yucheng Jiang, John Yang, and Diyi Yang. Collaborative gym: A framework for enabling and evaluating human-agent collaboration. In *International Conference on Learning Representations*, volume 2026, pp. 99616–99649, 2026.

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.

Weiwei Sun, Xuhui Zhou, Weihua Du, Xingyao Wang, Sean Welleck, Graham Neubig, Maarten Sap, and Yiming Yang. Training proactive and personalized llm agents. *arXiv preprint arXiv:2511.02208*, 2025.

Leitian Tao, Baolin Peng, Wenlin Yao, Tao Ge, Hao Cheng, Mike Hang Wang, Jianfeng Gao, and Sharon Li. Trace: Turn-level reward assignment via credit estimation for long-horizon agents. *arXiv preprint arXiv:2607.13988*, 2026.

Guoqing Wang, Sunhao Dai, Guangze Ye, Zeyu Gan, Wei Yao, Yong Deng, Xiaofeng Wu, et al. Information gain-based policy optimization: A simple and effective approach for multi-turn search agents. In *International Conference on Learning Representations*, volume 2026, pp. 104502–104529, 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.

Shirley Wu, Michel Galley, Baolin Peng, Hao Cheng, Gavin Li, Yao Dou, Weixin Cai, James Zou, Jure Leskovec, and Jianfeng Gao. Collabllm: From passive responders to active collaborators. *arXiv preprint arXiv:2502.00640*, 2025.

Ruihan Yang, Aili Chen, Xintao Wang, Jiangjie Chen, Siyu Yuan, Deqing Yang, Yanghua Xiao, et al. Aria: Training language agents with intention-driven reward aggregation. *Advances in Neural Information Processing Systems*, 38:109749–109781, 2026.

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*, 2022.

Michael Zhang, W Bradley Knox, and Eunsol Choi. Modeling future conversation turns to teach llms to ask clarifying questions. In *International Conference on Learning Representations*, volume 2025, pp. 60722–60742, 2025.

Tong Zhang, Peixin Qin, Yang Deng, Chen Huang, Wenqiang Lei, Junhong Liu, Dingnan Jin, Hongru Liang, and Tat-Seng Chua. Clamber: A benchmark of identifying and clarifying ambiguous information needs in large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 10746–10766, 2024.

Weikang Zhao, Xili Wang, Chengdi Ma, Lingbin Kong, Zhaohua Yang, Mingxiang Tuo, Xiaowei Shi, Yitao Zhai, and Xunliang Cai. Mua-rl: Multi-turn user-interacting agent reinforcement learning for agentic tool use. *arXiv preprint arXiv:2508.18669*, 2025.

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*, 2024.

Yifei Zhou, Song Jiang, Yuandong Tian, Jason Weston, Sergey Levine, Sainbayar Sukhbaatar, and Xian Li. Sweet-rl: Training multi-turn llm agents on collaborative reasoning tasks. *arXiv preprint arXiv:2503.15478*, 2025.

A CrossPO Pseudo Code

Algorithm 1 summarizes the full training procedure of CrossPO. Building on GRPO, CrossPO introduces request matching within each rollout group and credit assignment through aligned suffix-return contrasts. Requests are matched using a frozen embedding model, and the resulting credits are calibrated through standardization, shrinkage, and clipping. The calibrated credit augments the GRPO advantage at the first occurrence of each request class, while a fixed request cost is applied to every occurrence. These computations reuse the sampled trajectories, enabling finer-grained credit assignment without additional environment interactions or a learned critic.

Algorithm 1 CrossPO Training Procedure

Require: $\pi_\theta, \pi_{\text{ref}}, \mathcal{D}, G, e, \eta; \beta, \kappa, \sigma_{\min}, a_{\max}, c_0, \epsilon, \lambda_{\text{KL}}$

for each training iteration **do**
  $\theta_{\text{old}} \leftarrow \theta$
  for each x in a batch sampled from $\mathcal{D}$ **do**
    $\tau_i \sim \pi_{\theta_{\text{old}}}(\cdot \mid x), \quad r_i \leftarrow R(\tau_i), \quad i = 1, \dots, G$
    $\widehat{A}_i \leftarrow \dfrac{r_i - \text{mean}(r_1, \dots, r_G)}{\text{std}(r_1, \dots, r_G)}$ ▷ 0 if all rewards are equal
    $\widehat{A}^{\text{CrossPO}}_{i,k} \leftarrow \widehat{A}_i, \quad i = 1, \dots, G,\ k = 1, \dots, |o_i|$
    $\mathcal{E} \leftarrow \{i : \tau_i \text{ has no infrastructure failure}\}$
    Build $\mathcal{Q}_x$ on $\mathcal{E}$ using native IDs or complete linkage (e, η)
    for $q \in \mathcal{Q}_x$ **do**
      $\mathcal{I}_q \leftarrow \{i \in \mathcal{E} : \exists t,\ a_{i,t} \in C_q\}, \quad \mathcal{I}_{\neg q} \leftarrow \mathcal{E} \setminus \mathcal{I}_q$
      $n_q \leftarrow |\mathcal{I}_q|, \quad n_{\neg q} \leftarrow |\mathcal{I}_{\neg q}|, \quad A_q \leftarrow 0$
      $t_{i,q} \leftarrow \min\{t : a_{i,t} \in C_q\}, \quad i \in \mathcal{I}_q$
      if $n_q n_{\neg q} > 0$ **then**
        $G_{i,q} \leftarrow \displaystyle\sum_{u=t_{i,q}}^{T_i} r_{i,u}, \quad i \in \mathcal{I}_q$
        Compute $G_{j,\neg q}$ by Eq. (5), $\quad j \in \mathcal{I}_{\neg q}$
        $\widehat{\Delta}_q \leftarrow \dfrac{1}{n_q} \sum_{i \in \mathcal{I}_q} G_{i,q} - \dfrac{1}{n_{\neg q}} \sum_{j \in \mathcal{I}_{\neg q}} G_{j,\neg q}$
        $s_q \leftarrow \text{Std}\big((G_{i,q})_{i \in \mathcal{I}_q}, (G_{j,\neg q})_{j \in \mathcal{I}_{\neg q}}\big)$
        $n_{\text{eff},q} \leftarrow \dfrac{n_q n_{\neg q}}{n_q + n_{\neg q}}, \quad \rho_q \leftarrow \dfrac{n_{\text{eff},q}}{n_{\text{eff},q} + \kappa}$
        $A_q \leftarrow \text{clip}\left(\dfrac{\rho_q \widehat{\Delta}_q}{\max(s_q, \sigma_{\min})}, -a_{\max}, a_{\max}\right)$
      end if
      for $i \in \mathcal{I}_q,\ t :\ a_{i,t} \in C_q$ **do**
        $\widehat{A}^{\text{CrossPO}}_{i,k} \leftarrow \widehat{A}_i + \beta\,(\mathbf{1}\{t = t_{i,q}\} A_q - c_0)$ ▷ all $o_{i,k}$ in turn t
      end for
    end for
  end for
  Update θ by maximizing $J_{\text{CrossPO}}(\theta)$ ▷ fixed advantages and anchors
end for
return π_θ

B Theoretical Analysis

B.1 Reference-Time Averaging

The reference return should reflect the request times observed in the rollout group. Choosing just one of these times would make the comparison depend on an arbitrary selection. Equation (5) averages over all observed first-occurrence times, preserving their frequencies and removing this source of randomness.

Fix an anchor q with nonempty comparison groups. For $j \in \mathcal{I}_{\neg q}$, let $F_j(t)$ denote the return obtained by evaluating Eq. (5) at the single reference time t, including its termination and truncation rules. This is the quantity averaged to obtain $G_{j,\neg q}$.

Proposition B.1 (Averaging removes reference-time selection noise)**.** *Conditional on the observed rollouts, draw t uniformly from the list $(t_{i,q})_{i\in\mathcal{I}_q}$, counting repeated times separately. Then*

$$\begin{aligned}\mathbb{E}_t[F_j(t)] &= G_{j,\neg q},\\ \mathrm{Var}_t(F_j(t)) &= \frac{1}{n_q}\sum_{i\in\mathcal{I}_q}\big(F_j(t_{i,q}) - G_{j,\neg q}\big)^2.\end{aligned} \tag{12}$$

Using $G_{j,\neg q}$ preserves the expected reference return and removes the variance caused by selecting a reference time.

Proof. Each entry in the list is selected with probability $\frac{1}{n_q}$. Therefore,

$$\mathbb{E}_t[F_j(t)] = \frac{1}{n_q}\sum_{i\in\mathcal{I}_q} F_j(t_{i,q}) = G_{j,\neg q}.$$

Subtracting this mean from each possible return and averaging the squared differences gives the variance in Eq. (12). Evaluating all reference times leaves no random time selection, while retaining the same mean. □

This averaging aligns the two groups over the same distribution of starting times without adding environment rollouts. If rewards occur only at normal termination, with $r_{i,T_i} = R(\tau_i)$, every reference time gives $F_j(t) = R(\tau_j)$, including times beyond T_j. In that case, there is no reference-time selection noise to remove, and the contrast becomes the difference in mean episode rewards.

B.2 Return Contrast and Request Advantage

A request has positive advantage when its expected subsequent return exceeds the policy's average return at the same decision context. The return contrast provides evidence for this comparison, but uses trajectories that may have different preceding histories. The following analysis separates the effect of these differences from finite-sample error.

Write $\pi = \pi_{\theta_{\text{old}}}$. At a context (h, o), let $Q^\pi_q(h, o)$ and $Q^\pi_{\neg q}(h, o)$ be the expected subsequent returns when the current action is sampled from π conditional on being inside or outside C_q, respectively. Both use the reward sum in Eq. (4), and subsequent actions follow π. In particular, the second choice may still request q later in the episode. With $p_q(h, o) = \pi(C_q \mid h, o) \in (0, 1)$, the policy value is

$$V^\pi(h, o) = p_q(h, o)Q^\pi_q(h, o) + (1 - p_q(h, o))Q^\pi_{\neg q}(h, o).$$

Subtracting this average from the value of requesting q gives

$$\begin{aligned}A^\pi(h, o; q) &= Q^\pi_q(h, o) - V^\pi(h, o)\\ &= (1 - p_q(h, o))\big(Q^\pi_q(h, o) - Q^\pi_{\neg q}(h, o)\big).\end{aligned} \tag{13}$$

Thus the difference between the two expected returns has the same sign as the request-class advantage. When $R(\tau) = \sum_t r_t$, the same difference can be obtained from expected episode outcomes: at a fixed context, the current action cannot change rewards already accumulated, so their expected contribution cancels.

For independent trajectories drawn from the requesting and reference populations, define the expected return contrast

$$\Delta^\star_q = \mathbb{E}[G_{i,q} - F_j(t_{i,q}) \,|\, i \in \mathcal{I}_q,\ j \in \mathcal{I}_{\neg q}]\,. \tag{14}$$

This is the population quantity estimated by $\widehat{\Delta}_q$. To connect it to the comparison at (h, o), suppose the difference between the two comparisons is bounded by $b_q \geq 0$:

$$\big|\Delta^\star_q - \big(Q^\pi_q(h, o) - Q^\pi_{\neg q}(h, o)\big)\big| \leq b_q. \tag{15}$$

Here $b_q = 0$ means that the expected comparisons agree. A nonzero b_q allows for different histories and for the fact that reference trajectories never request q, whereas the alternative action at (h, o) may be followed by a later request.

Theorem B.1 (Request advantage from return comparisons)**.** *Fix C_q independently of the credit-estimation rollouts. Assume eligible rollouts are independent under π, with eligibility filtering preserving this law, and that both groups have positive probability. Suppose $G_{i,q}$ and $F_j(t)$ lie in a common interval of width $B > 0$ for all possible reference times. At a context (h, o) satisfying Eq.* (15) *and $0 < p_q(h, o) < 1$, define*

$$\varepsilon_q = B\sqrt{\frac{2}{n_{\mathrm{eff},q}}\log\frac{2}{\delta}}, \qquad 0 < \delta < 1. \tag{16}$$

Conditional on positive group sizes, with probability at least $1 - \delta$,

$$\Big|(1 - p_q(h, o))\widehat{\Delta}_q - A^\pi(h, o; q)\Big| \le (1 - p_q(h, o))(b_q + \varepsilon_q). \tag{17}$$

On this event, whenever $|\widehat{\Delta}_q| > b_q + \varepsilon_q$, the anchor credit satisfies

$$\mathrm{sign}(A_q) = \mathrm{sign}\big(A^\pi(h, o; q)\big). \tag{18}$$

Proof. We first bound the error in estimating $\Delta_q^\star$. Substituting the reference average into the contrast gives

$$\widehat{\Delta}_q = \frac{1}{n_q n_{\neg q}} \sum_{i \in \mathcal{I}_q} \sum_{j \in \mathcal{I}_{\neg q}} \big(G_{i,q} - F_j(t_{i,q})\big). \tag{19}$$

Conditional on the group memberships, each pair contains independent trajectories from the two populations. Every summand therefore has expectation $\Delta_q^\star$, giving $\mathbb{E}[\widehat{\Delta}_q \mid n_q, n_{\neg q}] = \Delta_q^\star$.

The summands share trajectories, so their number does not equal the number of independent observations. Replacing one requesting trajectory changes its own contribution by at most $\frac{B}{n_q}$. Its changed first-occurrence time also changes the reference averages, by at most another $\frac{B}{n_q}$ in total. Replacing one reference trajectory changes only its own averaged contribution, by at most $\frac{B}{n_{\neg q}}$. The bounded-differences inequality (McDiarmid et al., 1989) therefore gives

$$\mathbb{P}\Big(|\widehat{\Delta}_q - \Delta_q^\star| > u \,\Big|\, n_q, n_{\neg q}\Big) \le 2\exp\left(-\frac{2u^2}{B^2\left(\frac{4}{n_q} + \frac{1}{n_{\neg q}}\right)}\right). \tag{20}$$

Since

$$\frac{4}{n_q} + \frac{1}{n_{\neg q}} \le 4\left(\frac{1}{n_q} + \frac{1}{n_{\neg q}}\right) = \frac{4}{n_{\mathrm{eff},q}},$$

taking $u = \varepsilon_q$ bounds the failure probability by δ.

We next connect this sampling result to advantage. On the resulting event, Eq. (15) and the triangle inequality yield

$$\Big|\widehat{\Delta}_q - \big(Q_q^\pi(h, o) - Q_{\neg q}^\pi(h, o)\big)\Big| \le \varepsilon_q + b_q.$$

Multiplying by $1 - p_q(h, o)$ and using Eq. (13) proves Eq. (17). If $|\widehat{\Delta}_q| > b_q + \varepsilon_q$, the expected return difference has the same nonzero sign as $\widehat{\Delta}_q$. The positive shrinkage factor and normalization scale, followed by symmetric clipping, preserve this sign in A_q. □

The theorem gives a concrete condition for useful local credit: the observed contrast must exceed the combined effect of comparison mismatch and sampling error. More trajectories reduce the latter through $n_{\mathrm{eff},q}$; they do not by themselves remove differences between interaction histories.

The sampling result assumes a fixed request class, as with native request identifiers. Semantic clusters fitted to the same rollout group share data with credit estimation, so extending this guarantee to their selection requires accounting for that dependence. The reference-averaging identity in Section B.1 holds for any realized clustering.

B.3 Effect of Credit Calibration

Shrinkage and sample support. An imbalanced comparison gives a few trajectories considerable influence over one group mean. This effect is explicit with terminal-only rewards and normal termination, where the contrast is a difference of independent episode-reward means. If both populations have variance σ^2, then

$$\mathrm{Var}(\widehat{\Delta}_q \mid n_q, n_{\neg q}) = \frac{\sigma^2}{n_q} + \frac{\sigma^2}{n_{\neg q}} = \frac{\sigma^2}{n_{\mathrm{eff},q}}. \tag{21}$$

This explains why the smaller group matters even when the other group is large. For a fixed total count, balancing the groups maximizes $n_{\mathrm{eff},q}$; increasing imbalance lowers it and hence lowers ρ_q when $\kappa > 0$. For general aligned returns, the sampling bound in Theorem B.1 accounts for the shared reference times.

A smaller ρ_q suppresses sampling fluctuations, but also pulls the estimated contrast toward zero. The next result states when the variance reduction outweighs this bias.

Proposition B.2 (When shrinkage reduces estimation error)**.** *Under the sampling assumptions of Theorem B.1, condition all expectations and variances below on fixed positive group sizes. Then*

$$\mathbb{E}\big[(\rho_q\widehat{\Delta}_q - \Delta_q^\star)^2\big] = \rho_q^2\,\mathrm{Var}(\widehat{\Delta}_q) + (1-\rho_q)^2(\Delta_q^\star)^2. \tag{22}$$

For $\kappa > 0$, shrinking the contrast reduces or preserves its mean squared error exactly when

$$(\Delta_q^\star)^2 \le \frac{1+\rho_q}{1-\rho_q}\,\mathrm{Var}(\widehat{\Delta}_q). \tag{23}$$

Proof. At fixed group sizes, ρ_q is constant. Write

$$\rho_q\widehat{\Delta}_q - \Delta_q^\star = \rho_q(\widehat{\Delta}_q - \Delta_q^\star) - (1-\rho_q)\Delta_q^\star.$$

The first term has mean zero. Squaring and taking expectations therefore gives Eq. (22). The unshrunk estimator has mean squared error $\mathrm{Var}(\widehat{\Delta}_q)$. Subtracting it from Eq. (22) gives

$$(1-\rho_q)\left[(1-\rho_q)(\Delta_q^\star)^2 - (1+\rho_q)\,\mathrm{Var}(\widehat{\Delta}_q)\right],$$

which is nonpositive exactly under Eq. (23). □

Thus shrinkage is beneficial when sampling variability is large relative to the squared population contrast. The support-dependent factor reduces the influence of comparisons where either group is represented by few trajectories.

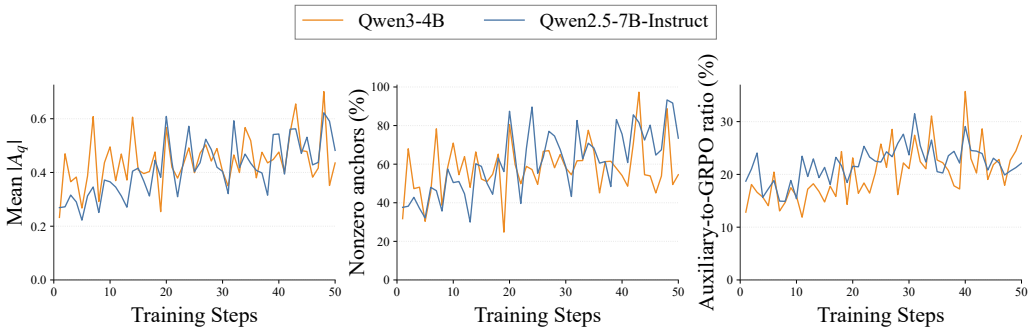
Normalization and the scale floor. Dividing by s_q expresses a contrast relative to the observed return variation. If this variation is very small, however, a small absolute contrast can still produce substantial standardized credit. The floor in Eq. (9) gives

$$|A_q| \le \min\left\{a_{\max}, \frac{\rho_q|\widehat{\Delta}_q|}{\sigma_{\min}}\right\}. \tag{24}$$

The first bound follows from clipping; the second follows from $\max(s_q, \sigma_{\min}) \ge \sigma_{\min}$ and the fact that clipping cannot increase magnitude. Consequently, as $\widehat{\Delta}_q$ tends to zero, the credit also tends to zero even if s_q vanishes at the same time. The scale floor therefore keeps negligible absolute differences from receiving large credit, while clipping limits the largest corrections. Both operations preserve the sign established by the return comparison.

C Training Details

Credit Dynamics. We examine how the action-level credit produced by CrossPO evolves during training on τ^2-Bench. As shown in Figures 5(a) and (b), both the mean absolute credit and the proportion of anchors receiving nonzero credit increase during the early training stage and subsequently remain relatively stable, despite natural step-level fluctuations. Together, these trends show that the



(a) Credit magnitude (b) Nonzero credit coverage (c) Relative strength of auxiliary credit

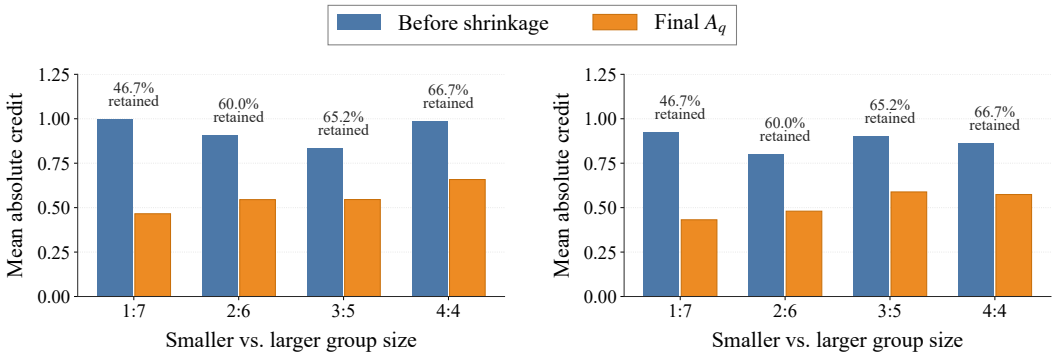
Figure 5: **Evolution of action-level credit during CrossPO training.** We track three credit statistics over the 50 training steps on τ^2-Bench using Qwen3-4B and Qwen2.5-7B-Instruct. (a) Mean absolute action-level credit $\mathbb{E}[|A_q|]$. (b) Percentage of anchors assigned nonzero credit. (c) Ratio of the mean absolute auxiliary credit correction introduced by CrossPO to the mean absolute GRPO advantage, computed over the same response-token positions.

Table 4: Key hyperparameters used for CrossPO training across the three benchmarks.

Hyperparameter	**UserGym**	**ColBench**	**τ^2-Bench**
Models and Optimization			
Semantic encoder	Qwen3-Embedding-0.6B	Qwen3-Embedding-0.6B	Qwen3-Embedding-0.6B
Actor learning rate	1×10^{-6}	1×10^{-6}	1×10^{-6}
Training batch size	64	32	32
Rollouts per prompt	8	8	8
PPO mini-batch size	32	32	32
Training schedule	5 epochs	5 epochs	50 steps
KL-loss coefficient	0.001	0.001	0.001
Actor thinking	Disabled	Disabled	Disabled
Maximum prompt length	1,152	2,048	8,192
Maximum response length	8,192	8,192	16,384
CrossPO Credit Estimation			
Embedding similarity threshold τ_s	0.70	0.70	0.70
Return source	Suffix return	Suffix return	Suffix return
Return standard-deviation floor $\sigma_{\min}$	0.1	0.1	0.1
Shrinkage coefficient κ	1.0	1.0	1.0
Credit clipping bound $a_{\max}$	1.25	1.25	1.25
Credit weight β	0.5	0.5	0.5
Information-request cost c_0	0.03	0.03	0.01

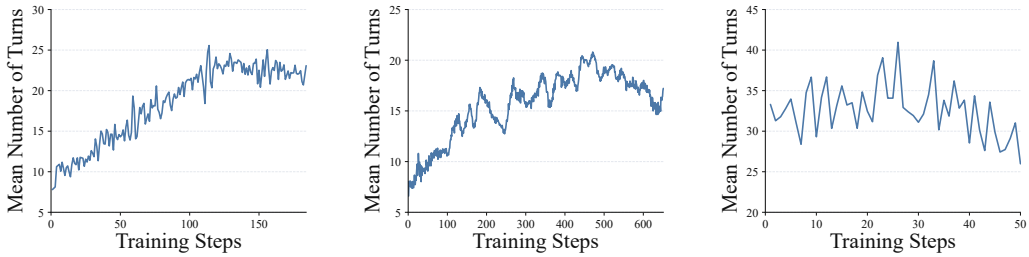
credit signal becomes active for a larger proportion of anchor actions during early training and remains relatively stable thereafter, rather than being confined to a small number of isolated anchors. Figure 5(c) further shows that the relative strength of the auxiliary credit initially increases and then stabilizes at a bounded level. Thus, the fine-grained credit provides a non-negligible training signal without dominating the underlying GRPO advantage. The broadly consistent dynamics across the two backbones further suggest that this behavior is not specific to a single model configuration.

Implementation Details. We train Qwen3-4B and Qwen2.5-7B-Instruct as the actor backbones and sample 8 trajectories for each prompt. Within each benchmark, CrossPO and all baselines use the same optimization settings, rollout budgets, and interaction limits. CrossPO groups semantically equivalent information requests using Qwen3-Embedding-0.6B with a similarity threshold of 0.70. The action-level credit is computed from suffix returns, standardized with a standard-deviation floor



(a) Qwen3-4B (b) Qwen2.5-7B-Instruct

Figure 6: **Effect of comparison-group support on small-sample shrinkage.** On τ^2-Bench, we group action-level credit estimates by the sizes of the trajectory subsets with and without the corresponding information action for (a) Qwen3-4B and (b) Qwen2.5-7B-Instruct. Blue bars show the mean absolute standardized return difference before shrinkage, orange bars show the resulting mean absolute credit A_q, and the annotations report the proportion retained after shrinkage.



(a) UserGym (b) ColBench (c) τ^2-Bench

Figure 7: **Training-time turn-length dynamics across benchmarks.** We report the mean number of turns per training trajectory for Qwen3-4B trained with CrossPO on (a) UserGym, (b) ColBench, and (c) τ^2-Bench.

of 0.1, adjusted using a shrinkage coefficient of $\kappa = 1$, and clipped to $[-1.25, 1.25]$. The resulting credit is incorporated into the GRPO advantage with weight $\beta = 0.5$. We use benchmark-specific information-request costs of 0.03, 0.03, and 0.01 for UserGym, ColBench, and τ^2-Bench, respectively. Differences in batch size, sequence length, and interaction horizon reflect the distinct protocols and trajectory lengths of the three benchmarks rather than method-specific optimization. Detailed hyperparameter settings are provided in Table 4.

As shown in Figure 6, each rollout group contains eight trajectories, and the horizontal axis characterizes the balance between the two subsets used for suffix-return comparison. Under the highly imbalanced 1:7 split, one side provides limited statistical support, and the shrinkage mechanism retains only 46.7% of the original magnitude. The retained proportion increases progressively to 66.7% as the two subsets become more balanced. This pattern is consistent across both backbones. These results show that CrossPO applies stronger attenuation to credit estimates derived from poorly supported comparisons while preserving a larger fraction of the signal when both subsets contain sufficient samples.

Figure 7 shows that the mean turn count follows distinct training dynamics across the three environments. On UserGym, it increases gradually during the first 120 training steps, subsequently stabilizes, and declines slightly after approximately step 130. On ColBench, the mean turn count increases over a longer training period before decreasing moderately toward the end. In contrast, τ^2-Bench exhibits a consistently higher turn count with substantial step-level variation during the 50 training steps. Overall, turn count depends on the task structure and optimization stage of each environment.

D CASE STUDIES

We present two paired ColBench trajectories, using Qwen3-4B agents, a GPT-4o-mini user simulator, and at most six user replies. Blue and green cards show agent turns and user replies; orange cards report saved evaluation results, and gray cards contain editorial notes. Dialogue excerpts and submitted code retain their recorded wording, with omitted turns marked. Evaluation results are not feedback provided to the agent.

D.1 CLARIFYING MISSING DECISION RULES

CrossPO elicits the category names and their horsepower boundaries before implementing the classifier. In the paired GRPO trace, repeated requests about unconfirmed low/medium/high categories do not obtain the missing rules.

Initial user request

Can you write a python function to categorize Saab car models based on their horsepower? The function should return the model type based on certain horsepower ranges, but I'm not sure what those ranges are.The signature of the function is: def classify_saab_car(horsepower)

CrossPO

Agent (Step 1)

What is the purpose of the function classify_saab_car(horsepower)?

User (Step 1)

The purpose of the function classify_saab_car(horsepower) is to categorize Saab car models based on their horsepower into specific types. It uses defined horsepower ranges to determine whether a model is classified as 'Linear', 'Aero', or 'Unknown'.

Agent (Step 2)

What are the specific horsepower ranges for the categories 'Linear', 'Aero', and 'Unknown'?

User (Step 2)

The horsepower ranges for the categories are as follows: 'Linear' for horsepower less than 200, 'Aero' for horsepower less than 240, and 'Unknown' for horsepower 240 and above. You can use these ranges to implement the function.

Agent (Step 3)

What is the return type of the function classify_saab_car(horsepower)?

User (Step 3)

The return type of the function classify_saab_car(horsepower) is a string, as it returns the model type based on the horsepower ranges. The possible return values are 'Linear', 'Aero', or 'Unknown'.

Agent (Step 4): Final answer

```
def classify_saab_car(horsepower):
    if horsepower < 200:
        return 'Linear'
    elif horsepower < 240:
        return 'Aero'
    else:
        return 'Unknown'
```

Recorded evaluation

Test pass fraction: 1.000

User calls: 3 / 6

GRPO

Agent (Step 1)

Sure, can you please tell me what the specific horsepower ranges are for categorizing the Saab car models? For example, what is the range for low, medium, and high horsepower?

User (Step 1)

I don't know the specific horsepower ranges for categorizing Saab car models. However, I can provide a function that classifies them based on certain ranges I have.

Agent (Step 2)

Sure, could you please provide the specific ranges for low, medium, and high horsepower that you want to use for classification? For example, what is the minimum and maximum horsepower for each category?

User (Step 2)

I don't have the specific horsepower ranges for categorizing Saab car models. However, I can provide a function that classifies them based on certain ranges I have.

Transcript note

Steps 3–5 omitted: the agent repeats its Step 2 question verbatim in all three turns. The user replies continue to say that the ranges are not known.

Agent (Step 6): Final answer

```
def classify_saab_car(horsepower):
    if horsepower < 100:
        return "low"
    elif 100 <= horsepower < 200:
        return "medium"
    else:
        return "high"
```

Recorded evaluation

Test pass fraction: 0.000

User calls: 5 / 6

CrossPO uses the supplied labels and 200/240 thresholds; GRPO submits unconfirmed labels and cutoffs. CrossPO still asks about the return type, but avoids GRPO's repeated unresolved request.

D.2 Acting on Sufficiently Specified Requirements

The initial request already specifies consecutive duplicates, arbitrary item types, and a new output list. CrossPO submits code immediately, whereas GRPO makes four user calls, including requests that revisit the permitted item types and output behavior. For GRPO, we show exchanges 2–3 and the final submitted code; exchanges 1 and 4 are omitted.

Initial user request

I want the AI to write a python function that removes consecutive duplicates in a list. I have a list of items and I want a new list with all the consecutive duplicates removed. The function should work with any type of item in the list. Can you help me figure out how to define this function?The signature of the function is: def compress(lst)

CrossPO (0 user calls)

Agent (Step 1): Final answer

```
def compress(lst):
    if not lst:
        return []
    result = [lst[0]]
    for item in lst[1:]:
        if item != result[-1]:
            result.append(item)
    return result
```

Recorded evaluation

Test pass fraction: 1.000

User calls: 0 / 6

GRPO (4 user calls)

Agent (Step 2)

Sure, can you explain how the function should handle the case where the list has multiple consecutive duplicates? For example, if the input is [1, 1, 1, 2, 2, 3, 3, 3], what would the expected output be?

Also, what data types are allowed in the list? Are there any specific constraints on the items in the list?

User (Step 2)

The expected output for the input [1, 1, 1, 2, 2, 3, 3, 3] would be [1, 2, 3]. The function should work with any type of item in the list, without specific constraints.

Agent (Step 3)

Sure, can you confirm how the function should handle the case where the list has different data types? For example, if the input is [1, 'a', 'a', 2], what would the expected output be?

Also, what is the expected return type of the function? Should it return a new list or modify the original list in place?

User (Step 3)

The expected output for the input [1, 'a', 'a', 2] would be [1, 'a', 2]. The function should return a new list without modifying the original list.

Agent (Step 5): Final answer

```
def compress(lst):
    if not lst:
        return []
    result = [lst[0]]
    for i in range(1, len(lst)):
        if lst[i] != lst[i-1]:
            result.append(lst[i])
    return result
```

Recorded evaluation

Test pass fraction: 1.000

User calls: 4 / 6

Both submissions achieve a test pass fraction of 1.000. CrossPO completes the task without further user interaction, while the displayed GRPO exchanges revisit information already provided in the initial request.