ACRE: ACTION-LEVEL CREDIT ASSIGNMENT WITH RUBRIC EVALUATION FOR LONG-HORIZON AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Long-horizon agentic tasks require reinforcement learning (RL) methods to optimize both local decisions and final task completion. While group-based RL methods provide an effective framework for learning from sparse outcome rewards, it remains difficult to determine which intermediate actions contribute to success or failure. Recent stepwise policy optimization methods improve credit assignment by introducing intermediate signals, but these signals are still derived from downstream consequences, leaving local decision quality entangled with future outcomes. We introduce **A**ction-level **C**redit Assignment with **R**ubric **E**valuation for Long-Horizon Agents (ACRE). Rather than estimating process advantages from outcome-derived returns, ACRE uses reusable rubrics for different functional action families to turn semantic evaluations of intermediate actions into process rewards. These rewards are mean-centered within matched environment-state groups to form relative process advantages, which are combined with the episode-level outcome advantage for policy optimization. In this way, ACRE provides a direct step-level learning signal while preserving episode-level supervision for task completion. Across ALFWorld and WebShop with Qwen2.5-1.5B/3B/7B-Instruct, ACRE consistently improves over return-based process advantage estimation. We further show that rubric-based outcome evaluation can approach verifiable environment rewards, suggesting a practical direction for agentic RL in more open-ended settings where deterministic outcome verification is unavailable.

1 INTRODUCTION

Large language models (LLMs) are increasingly deployed as agents that interact with external environments through multi-step reasoning and tool use (Yao et al., 2023; Shinn et al., 2023). These agentic tasks unfold as long-horizon sequences of observations, reasoning, actions, and environment feedback, creating a fundamental credit assignment challenge for reinforcement learning (RL). This challenge naturally motivates two complementary forms of supervision: process supervision for *local decision quality* and outcome supervision for *global task completion* (Lightman et al., 2024; Sutton & Barto, 2018). However, agent environments often provide only sparse outcome rewards, so a good action can be followed by a failure trajectory, and a bad action can be followed by a successful recovery. More recently, long-horizon agentic RL has introduced finer-grained credit assignment (Feng et al., 2025; He et al., 2026; Cheng et al., 2026; Zhang et al., 2026). But these process signals are still commonly derived from *post-decision consequences*.

In this work, we take a different perspective and construct process supervision directly by evaluating each step against its *pre-decision context*. For example, in a shopping task, even without observing subsequent actions or the terminal outcome, a repeated option selection can be identified as redundant, while a choice that violates an observed constraint can be identified as inappropriate. Based on this observation, we define the process reward of an action as its quality under the interaction history available at that decision. However, obtaining such step-level evaluations is nontrivial. The criteria for a good action depend on both the current context and the action's functional role: selecting an option may require checking user constraints, whereas a navigation action may be judged by whether it moves the agent toward relevant information. Training a process reward model to capture these distinctions can require additional process annotations and task-specific training (Chae et al., 2025). Instead, we can make the relevant criteria explicit in rubrics and use an LLM judge to evaluate each action against them in its pre-decision context (Gunjal et al., 2026).

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.

We therefore introduce ACRE (**A**ction-level **C**redit Assignment with **R**ubric **E**valuation for Long-Horizon Agents). For a given agent environment, ACRE organizes actions by their functional roles and defines a set of reusable rubrics for each action family. At each interaction step, a frozen LLM judge evaluates the current action using its interaction history and matched rubrics, producing an immediate process reward. The reward is then mean-centered against those of actions observed under the same environment state to obtain a relative process advantage. Finally, the step-level process advantage is combined with the episode-level outcome advantage, allowing local decision signals and global outcome signals to provide complementary supervision during policy optimization.

We evaluate ACRE on WebShop (Yao et al., 2022) and ALFWorld (Shridhar et al., 2021) across three Qwen2.5 (Yang et al., 2025) model scales. ACRE consistently outperforms return-based process credit, with maximum success-rate improvements of 6.5 and 15.2 percentage points on WebShop and ALFWorld, respectively. Additional analyses demonstrate that direct process rewards provide more faithful assessments of local action quality, and ablations confirm the importance of action-specific rubric selection and decision-time evaluation context. Finally, we show that rubric-based outcome supervision can achieve comparable performance on WebShop, suggesting the potential of semantic rewards for open-ended agentic tasks where complete programmatic verification is unavailable.

Our contributions are summarized as follows:

- We formulate long-horizon agentic credit assignment from two complementary perspectives: *local decision quality* under the information available at decision time and *global task completion* reflected by eventual trajectory outcomes.
- We introduce ACRE, which directly evaluates intermediate actions with reusable action-specific rubrics and converts these semantic rewards into mean-centered state-conditioned relative process advantages, while retaining episode-level supervision for global task completion.
- We provide analytical characterization of mean-centered direct local credit, demonstrating and empirically show that direct process rewards better reflect intermediate action quality and improve long-horizon task performance across environments and model scales.

2 Related Work

Process supervision for LLM agents. Providing informative supervision for intermediate agent steps has been explored through both process reward models (PRMs) and general semantic evaluators. Web-Shepherd trains a specialized PRM from step-level preferences and structured checklists for web navigation (Chae et al., 2025), while AgentPRM models the promise and progress of agent decisions using value-based supervision (Xi et al., 2026). Rubric-based evaluation offers another form of process supervision by expressing desired behavior through explicit semantic criteria. Rubrics as Rewards (RaR) uses instance-specific rubrics as reward signals for on-policy RL beyond fully verifiable domains (Gunjal et al., 2026). More recent agentic methods use rubrics at different granularities: ARCO constructs step-specific rubrics with a rubric-conditioned evaluator (Tian et al., 2026), TRCA evaluates action-induced transitions with rubric criteria for execution and progress (Zhang et al., 2026), and DRACO dynamically evaluates trajectory-level rubrics before redistributing their credit to responsible steps (Gandhi et al., 2026). In contrast, ACRE uses reusable action-specific rubrics to evaluate each action from its pre-decision context, directly providing semantic supervision for local action quality.

Fine-grained credit assignment. Long-horizon agentic RL increasingly augments trajectory-level outcome supervision with finer-grained process credit. GiGPO constructs state-conditioned action groups and estimates step-level advantages from discounted future returns (Feng et al., 2025), while HGPO refines such grouping using historical context (He et al., 2026). GraphGPO exploits state-transition structure to estimate progress toward task completion (Cheng et al., 2026). More recently, BEACON partitions long trajectories around intermediate milestones and combines segment-level with trajectory-level advantages to reduce credit misattribution from distant failures (Wang et al., 2026), while TRACE derives turn-level rewards from temporal-difference changes in state values for long-horizon tool-use agents (Tao et al., 2026). These approaches primarily infer intermediate credit from downstream outcomes, state transitions, or estimated future progress. ACRE instead separates decision-time local quality from eventual task utility.

3 PROBLEM FORMULATION

3.1 LONG-HORIZON AGENTIC RL

Let $x \sim p(\mathcal{X})$ denote a task instruction and π_θ an LLM-based policy. At interaction step t, the environment provides a state s_t, while the policy acts on the full pre-decision context $h_t = (x, s_1, a_1, \ldots, s_t)$, which contains the information available before the current action is chosen. The agent then generates $a_t \sim \pi_\theta(\cdot \mid h_t)$. A trajectory is denoted by $\tau = \{(s_t, a_t, r_t)\}_{t=1}^{T}$, where r_t is the environment reward and T is the interaction horizon.

In long-horizon agent benchmarks, environment rewards are often sparse or delayed and may be issued only at task termination. In the terminal-reward setting, $r_t = 0$ for $t < T$, so the trajectory return $R(\tau) = \sum_{t=1}^{T} r_t$ reduces to the terminal outcome reward r_T. This scalar provides supervision for overall task completion, but does not by itself specify the quality of individual decisions made along the trajectory.

3.2 GROUP-RELATIVE CREDIT ASSIGNMENT

Group Relative Policy Optimization (GRPO) estimates policy advantages through relative comparison within a group of sampled trajectories, avoiding an explicit value model (Shao et al., 2024). For a task x, let $G^E(x) = \{\tau^{(1)}, \ldots, \tau^{(N)}\}$ denote a group of trajectories sampled from the current policy. Given an outcome reward $r(\tau)$, the episode-level relative advantage of trajectory τ is

$$A^E(\tau) = \frac{r(\tau) - \text{mean}\left(\{r(\tau') : \tau' \in G^E(x)\}\right)}{F_{\text{norm}}\left(\{r(\tau') : \tau' \in G^E(x)\}\right)}, \tag{1}$$

where $F_{\text{norm}}(\cdot)$ is the group standard deviation when normalization is enabled and 1 otherwise.

For intermediate actions, meaningful relative credit requires comparing decisions made under comparable local conditions. GiGPO (Feng et al., 2025) therefore groups action occurrences according to their pre-decision environment state, with $G^S(s) = \{a_t \mid s_t = s\}$ collected across sampled trajectories. This state-conditioned grouping provides a practical comparison set for estimating process advantages among alternative decisions encountered under the same environment state.

3.3 LIMITATION OF OUTCOME-DERIVED PROCESS CREDIT

Once comparable action groups are available, the central question becomes what quantity should be compared within each group. Existing group-based agentic RL methods commonly derive this quantity from downstream trajectory outcomes. For an action taken at step t, its discounted return-to-go is $R(a_t) = \sum_{k=t}^{T} \gamma^{k-t} r_k$, where $\gamma \in (0, 1]$ is the discount factor. GiGPO (Feng et al., 2025) estimates the corresponding process advantage by comparing these returns within the state-conditioned group

$$A^S_{\text{return}}(a_t) = \frac{R(a_t) - \text{mean}\left(\{R(a') \mid a' \in G^S(s_t)\}\right)}{F_{\text{norm}}\left(\{R(a') \mid a' \in G^S(s_t)\}\right)}. \tag{2}$$

Return-to-go captures the downstream consequences associated with an action, but it does not in general identify the quality of that action at the time it is chosen. This distinction becomes explicit under terminal-only feedback.

Proposition 1 (Limited local identifiability from terminal returns)**.** *Suppose the environment provides only a terminal reward Y, such that $r_t = 0$ for $t < T$ and $r_T = Y$. Then*

$$R(a_t) = \gamma^{T-t} Y. \tag{3}$$

Consequently, if $Y = 0$, all intermediate actions receive the same zero return; if $Y > 0$ and $0 < \gamma < 1$, the return increases with temporal proximity to the terminal outcome; and if $\gamma = 1$, all actions receive the same return Y. Thus, terminal return-to-go is determined by the eventual outcome and temporal position, and does not in general identify local decision quality.

The proof is provided in Appendix A.1. This result does not imply that return-based credit is uninformative: return-to-go captures downstream utility and remains important for long-horizon task

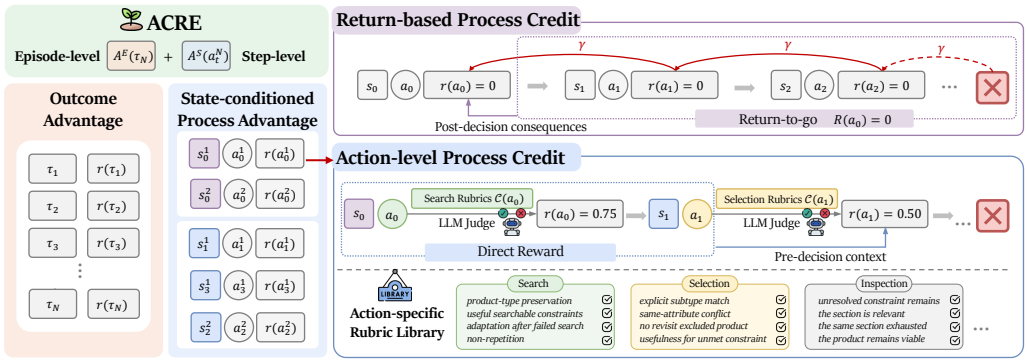


Figure 1: Overview of ACRE. Episode-level outcome advantages provide global supervision, while state-conditioned action groups enable relative process credit under the same state. The upper branch illustrates return-based process credit from post-decision consequences. The lower branch shows the direct process-credit branch used in ACRE: each action is matched to an action-specific rubric and evaluated by a frozen LLM judge using only the pre-decision context to produce an immediate process reward. The resulting step-level process advantage is then combined with the episode-level advantage for policy optimization.

optimization. Rather, its limitation concerns *local identifiability*: actions of different decision-time quality can receive identical returns, while differences in return can arise from later decisions or temporal distance to the outcome, reflecting the classical delayed-credit problem (Arjona-Medina et al., 2019). This motivates a complementary question: *can process credit be constructed directly from the quality of an action under its pre-decision context, rather than inferred solely from downstream outcomes?* Section 4 develops ACRE from this perspective.

4 METHOD: ACRE

ACRE augments episode-level outcome supervision with a separate source of credit for decision-time local quality. Its key idea is to decouple *how local action quality is evaluated* from *how that evaluation is converted into process credit*. Specifically, we first use action-specific rubrics to assign an immediate semantic reward to each action under its pre-decision context, and then directly compare these rewards within state-conditioned action groups to obtain relative process advantages. The resulting local credit is combined with the episode-level outcome advantage during policy optimization. Figure 1 summarizes this formulation and contrasts it with return-based process credit.

4.1 ACTION-SPECIFIC RUBRIC EVALUATION

To obtain decision-time supervision for local action quality, ACRE uses a frozen LLM judge with reusable action-specific rubrics. The evaluation consists of three steps.

1. Action-conditioned rubric selection. We partition the action space into functional families and associate each family with a fixed set of evaluation criteria. For an action a_t, let $\mathcal{C}(a_t) = \{c_m\}_{m=1}^{M_t}$ denote the rubric selected according to its action family. This design allows different types of actions to be evaluated according to their functional roles, while reusing the same family-level criteria across tasks and trajectories. The rubrics remain fixed within each matched training run, and their construction and complete environment-specific definitions are provided in Appendix B.1.

2. Decision-time semantic evaluation. At step t, the frozen judge receives the pre-decision context h_t, the selected action a_t, and its matched rubric $\mathcal{C}(a_t)$. For each criterion c_m, it produces a judgment $\ell_{t,m} \in \{\text{PASS}, \text{FAIL}, \text{N/A}\}$ together with supporting evidence. Crucially, the judge does not observe subsequent actions or the terminal outcome, so the evaluation reflects only the quality of the current decision under information available at that time. The complete judge interface is given in Appendix B.2.

3. Structured reward aggregation. We deterministically aggregate the applicable criterion-level judgments into an immediate process reward:

$$r^S(a_t) = \frac{\sum_{m=1}^{M_t} \mathbf{1}[\ell_{t,m} = \textsc{Pass}]}{\sum_{m=1}^{M_t} \mathbf{1}[\ell_{t,m} \neq \text{N/A}]}. \tag{4}$$

The score is the mean over applicable criteria unless the action violates a deterministically checkable constraint or a rubric-designated hard criterion, in which case $r^S(a_t) = 0$. If all criteria are N/A, no process score is assigned. This aggregation separates semantic evaluation from deterministic constraint enforcement while using only information available at decision time. The resulting $r^S(a_t)$ therefore provides a structured assessment of local decision quality independent of downstream trajectory outcomes.

4.2 Direct Process Advantage and Local–Global Supervision

Given the decision-time reward $r^S(a_t)$, ACRE converts it directly into a mean-centered group-relative process advantage using the state-conditioned groups introduced in Section 3.2. For an action a_t, we compute

$$A^S(a_t) = r^S(a_t) - \text{mean}\left(\left\{r^S(a') \mid a' \in G_S(s_t)\right\}\right), \tag{5}$$

Unlike the return-based estimator in Equation 2, ACRE applies this relative comparison directly to decision-time rewards without accumulating supervision from subsequent actions. The resulting $A^S(a_t)$ measures how much the current action's local quality deviates from other decisions encountered under the same environment state. This mean-centered formulation preserves the relative ordering and reward differences among locally comparable actions, while removing only the group-level offset. We use $A^S(a_t)$ as a group-relative process advantage, rather than as an estimator of the classical long-horizon advantage $Q^\pi(s,a) - V^\pi(s)$. Appendix A.2 provides a policy-gradient interpretation of this construction and discusses the difference between idealized same-context grouping and the practical state-conditioned estimator.

Proposition 1 motivates introducing such a local signal by showing that terminal returns do not generally identify decision-time action quality. The local signal, however, is intended to complement rather than replace supervision for eventual task completion. We therefore combine it with the episode-level outcome advantage

$$A^{\text{ACRE}}(a_t) = \alpha_E A^E(\tau) + \alpha_S A^S(a_t). \tag{6}$$

where α_E and α_S control the contributions of global outcome supervision and local process supervision. We set $\alpha_E = \alpha_S = 1$ in the main experiments. The resulting advantage is optimized with the token-level clipped objective described in Appendix C.1.

The two components in Equation 6 provide complementary views of an action: $A^E(\tau)$ reflects the eventual outcome of the trajectory containing the action, whereas $A^S(a_t)$ assesses its local quality using information available when it was chosen. Combining them allows policy optimization to account for both eventual task completion and the quality of decisions along the way. The local term is not intended to estimate the long-horizon advantage or reproduce the ordering induced by downstream returns.

4.3 Extension: Rubric-Based Outcome Supervision

The main formulation uses the environment reward for episode-level supervision. We additionally consider whether the same structured semantic evaluation paradigm can provide the global signal when a complete programmatic verifier is unavailable. Given a completed trajectory τ, an outcome-level rubric judge evaluates whether the task requirements are satisfied and produces a terminal reward $r_{\text{rub}}(\tau)$. This reward replaces the official environment reward when computing the episode-level advantage $A^E(\tau)$ in Equation 1, while the action-level reward $r^S(a_t)$ and direct process advantage $A^S(a_t)$ remain unchanged.

For WebShop, we combine semantic requirement judgments with deterministic checks for verifiable conditions and assign full reward to complete success while retaining limited credit for valid partial matches. The exact outcome rubric and reward construction are provided in Appendix B.3. Section 5.4 evaluates this extension.

Table 1: Main results on ALFWorld and WebShop. Values are reported as mean with standard deviation over three fixed evaluation seeds for the selected checkpoint. Bold denotes the best performance for each model and metric.

Methods	ALFWorld							WebShop	
	Pick	Look	Clean	Heat	Cool	Pick2	All	Score	SR
Qwen2.5-1.5B-Instruct									
Base	$1.03_{(0.90)}$	$0.00_{(0.00)}$	$0.00_{(0.00)}$	$1.19_{(2.06)}$	$0.00_{(0.00)}$	$0.72_{(1.26)}$	$0.52_{(0.23)}$	$46.69_{(4.08)}$	$7.55_{(1.08)}$
GRPO	$80.50_{(6.07)}$	$49.24_{(10.90)}$	$60.15_{(7.05)}$	$49.53_{(10.65)}$	$51.88_{(5.46)}$	$35.63_{(7.68)}$	$56.77_{(2.77)}$	$84.52_{(1.37)}$	$74.67_{(2.90)}$
GiGPO	$87.30_{(1.76)}$	$65.44_{(9.04)}$	$75.60_{(4.62)}$	$71.98_{(7.16)}$	$82.51_{(0.66)}$	$58.57_{(6.28)}$	$75.13_{(2.88)}$	$85.11_{(2.10)}$	$76.69_{(3.54)}$
ACRE	$\mathbf{97.26}_{(1.26)}$	$\mathbf{86.84}_{(6.20)}$	$\mathbf{92.50}_{(5.19)}$	$\mathbf{92.11}_{(8.57)}$	$\mathbf{88.30}_{(6.18)}$	$\mathbf{80.25}_{(1.81)}$	$\mathbf{90.36}_{(2.15)}$	$\mathbf{87.18}_{(1.17)}$	$\mathbf{78.84}_{(2.60)}$
Qwen2.5-3B-Instruct									
Base	$25.33_{(2.83)}$	$32.95_{(12.02)}$	$3.02_{(3.05)}$	$0.98_{(1.70)}$	$0.00_{(0.00)}$	$0.57_{(1.00)}$	$9.90_{(1.37)}$	$17.65_{(1.55)}$	$1.82_{(0.49)}$
GRPO	$87.99_{(0.93)}$	$82.96_{(1.12)}$	$52.30_{(8.81)}$	$45.34_{(9.77)}$	$30.99_{(9.65)}$	$22.64_{(2.64)}$	$53.26_{(4.29)}$	$86.36_{(2.51)}$	$77.41_{(3.18)}$
GiGPO	$95.71_{(2.58)}$	$83.76_{(4.37)}$	$87.99_{(5.95)}$	$71.58_{(13.46)}$	$86.42_{(6.75)}$	$60.63_{(5.25)}$	$82.16_{(2.54)}$	$87.35_{(2.10)}$	$80.08_{(2.56)}$
ACRE	$\mathbf{97.53}_{(2.30)}$	$\mathbf{88.32}_{(5.83)}$	$\mathbf{91.40}_{(6.29)}$	$\mathbf{81.25}_{(9.22)}$	$\mathbf{94.29}_{(1.01)}$	$\mathbf{73.80}_{(11.03)}$	$\mathbf{89.19}_{(0.81)}$	$\mathbf{89.29}_{(1.66)}$	$\mathbf{81.97}_{(2.93)}$
Qwen2.5-7B-Instruct									
Base	$40.79_{(4.43)}$	$30.17_{(7.63)}$	$4.76_{(1.63)}$	$6.64_{(3.27)}$	$6.53_{(4.32)}$	$15.37_{(2.61)}$	$18.36_{(2.95)}$	$23.66_{(0.19)}$	$4.30_{(0.70)}$
GRPO	$98.81_{(2.06)}$	$95.48_{(5.07)}$	$97.75_{(2.23)}$	$92.56_{(4.74)}$	$96.53_{(1.29)}$	$33.38_{(1.43)}$	$84.77_{(1.35)}$	$87.47_{(0.61)}$	$77.60_{(1.19)}$
GiGPO	$99.52_{(0.82)}$	$95.75_{(4.62)}$	$99.02_{(1.70)}$	$92.87_{(3.49)}$	$\mathbf{96.63}_{(2.92)}$	$58.49_{(9.19)}$	$90.23_{(1.03)}$	$87.89_{(2.07)}$	$78.97_{(4.01)}$
ACRE	$\mathbf{100.00}_{(0.00)}$	$\mathbf{100.00}_{(0.00)}$	$\mathbf{100.00}_{(0.00)}$	$\mathbf{96.70}_{(3.50)}$	$94.44_{(0.86)}$	$\mathbf{78.36}_{(3.20)}$	$\mathbf{94.66}_{(0.81)}$	$\mathbf{91.84}_{(1.49)}$	$\mathbf{85.48}_{(2.76)}$

5 EXPERIMENTS

5.1 EXPERIMENTAL SETUP

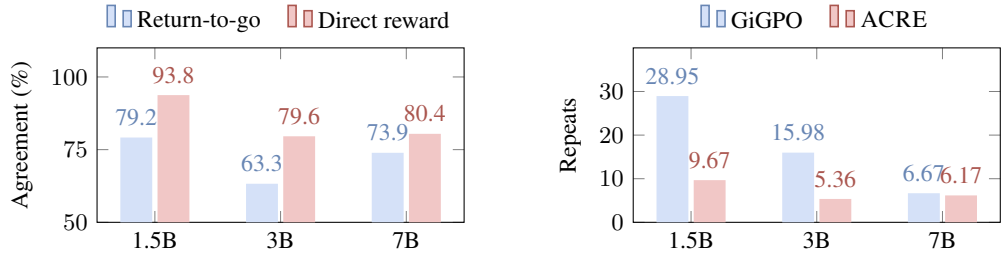
Benchmarks. We evaluate ACRE on two representative long-horizon agent benchmarks, **WebShop** (Yao et al., 2022) and **ALFWorld** (Shridhar et al., 2021). WebShop requires an agent to search, inspect, and purchase a product satisfying a natural-language instruction, while ALFWorld evaluates text-based interaction across six categories of embodied household tasks. For rubric evaluation, we map their nonterminal action spaces into seven and ten functional action families, respectively. The complete action taxonomy and family-specific rubrics are provided in Appendix B. We report task Score and success rate (SR) on WebShop and SR on ALFWorld.

Baselines. We evaluate Qwen2.5-1.5B/3B/7B-Instruct (Yang et al., 2025) and compare ACRE against the base instruction-tuned policy, **GRPO** (Shao et al., 2024), and **GiGPO** (Feng et al., 2025). GRPO uses trajectory-level group-relative supervision, while GiGPO additionally estimates process advantages from return-to-go within state-conditioned action groups. ACRE retains the same state-conditioned grouping but estimates process advantages directly from action-level rubric rewards. All main RL comparisons use the same official environment outcome reward for episode-level supervision, isolating the effect of process supervision and advantage estimation.

Implementation Details. All RL methods are implemented under the same agent-training framework and use matched policy context, task sampling, rollout budgets, and optimization budgets within each environment. ACRE uses a frozen, locally served Qwen3.5-35B-A3B (Qwen Team, 2026) model as the rubric judge. We evaluate each selected checkpoint on three fixed environment-seeded task streams and report the mean and sample standard deviation across them. Full training hyperparameters, hardware configuration, checkpoint selection, and evaluation protocol are provided in Appendix C, while rubric definitions and judge interfaces are detailed in B.2.

5.2 MAIN RESULTS

Overall task performance. Table 1 reports the main results across two long-horizon environments and three model scales. ACRE consistently outperforms GiGPO in success rate across all settings, indicating that the benefit of direct local process supervision is not restricted to a particular model scale or environment. The largest improvements are observed in the more challenging ALFWorld setting, where ACRE improves Qwen2.5-1.5B-Instruct by 15.23 percentage points in SR over GiGPO, suggesting that explicit local credit is particularly beneficial when outcome-based supervision provides limited guidance. On WebShop, ACRE also improves both task Score and SR, with the largest SR gain reaching 6.51 points. These results support that direct decision-time process credit complements outcome supervision and provides more effective learning signals for long-horizon agent optimization.



(a) Local-ranking agreement **(b) Repeated rollout actions**

Figure 2: Analysis of process signals on WebShop. **(a)** Local-ranking agreement between process rewards and consensus action preferences. **(b)** Frequency of repeated actions after training.

Direct rewards better reflect local decision quality. We examine whether the process signal better captures the quality of individual decisions. We sample 150 WebShop action pairs from matched decision contexts for which both direct reward and return-to-go assign different scores, allowing each signal to produce a strict pairwise ranking. GPT-5.6 Sol (OpenAI, 2026), Claude Sonnet 4.6 (Anthropic, 2026), and DeepSeek V4 Pro (DeepSeek-AI, 2026) independently judge the locally preferable action using only information available at decision time. The annotation protocol and full prompt are provided in Appendix D.1. Among the 143 pairs with a strict consensus preference, direct rubric rewards agree with the reference in 84.6% of cases, compared with 72.0% for return-to-go. This improvement holds across all three Qwen2.5 policy sizes in Figure 2(a). Under this independent reference, direct rubric rewards more faithfully rank actions by decision-time quality than return-to-go. Figure 3 shows three representative settings in which downstream return does not isolate local decision quality: redundant actions followed by eventual success, appropriate actions followed by later failure, and poor actions followed by subsequent recovery. Appendix D.5 provides additional recorded cases from WebShop and ALFWorld.

Complementary credit during training. We next examine how direct process credit complements episode-level outcome supervision during training. The analysis protocol for identifying complementary credit regimes and estimating downstream utility is described in Appendix D.3. Across training steps 1–150, 38.14% of all WebShop trajectory groups and 35.32% of the analyzed ALFWorld groups fall into a *process-only* regime, where the episode-level outcome advantage is zero but at least one state-conditioned process group still provides non-zero relative credit. In contrast, *outcome-only* groups account for only 3.96% of WebShop groups and 0.02% of the analyzed ALFWorld groups. Direct process credit therefore frequently supplies relative supervision in trajectory groups where the outcome branch provides no distinction among sampled trajectories.

We further find that this additional process credit does not simply reproduce outcome-derived credit. At the initial Qwen2.5-3B policy checkpoint, where rubric scores remain discriminative across candidate actions, direct process credit has nearly zero cosine similarity with rollout-estimated downstream utility (-0.005), whereas return-to-go exhibits strong positive cosine similarity (0.655). Together with the local-quality ranking analysis above, this suggests that direct process credit provides decision-time supervision that captures information complementary to downstream outcome credit.

Reduced redundant behavior. Finally, Figure 2(b) shows that ACRE substantially reduces exact repeated actions on WebShop for the 1.5B and 3B policies, with reductions of approximately two thirds relative to GiGPO. The difference is smaller at 7B. The repeated-step rate shows the same overall pattern in Appendix D.4. This behavior is consistent with direct process supervision penalizing locally unproductive decisions even when later actions can recover from them.

5.3 Ablation Studies

Our ablations suggest a simple recipe for applying ACRE effectively: match the evaluation criteria to the functional role of the current action, judge the action only from information available when the decision is made, and convert the resulting local reward directly into a relative process advantage rather than re-accumulating future rubric scores. Together, these choices preserve the locality of the process signal from reward construction to advantage estimation. We study these design choices on

Case A: Redundancy despite success

State. The request specifies `x-large` and `white`; the product offers both.
Actions. `click[x-large]` → `click[x-large]` again.

Direct reward $r(a)$	$0.8 \rightarrow 0.4$	**Return-to-go** $R(a)$	$8.5738 \rightarrow 9.0250$

The second selection repeats an option already set, but is closer to the successful purchase.
Continuation. Select white → buy. **Outcome:** Success

Case B: Local correctness despite eventual failure

State. The request specifies `bbq`; the product offers BBQ and other flavors.
Actions. `click[bbq]` → `click[sour cream & onion]`.

Direct reward $r(a)$	$0.8 \rightarrow 0.0$	**Return-to-go** $R(a)$	$0.0000 \rightarrow 0.0000$

The first choice satisfies the flavor constraint; the second undoes it, while their returns are tied.
Continuation. Buy with the conflicting flavor. **Outcome:** Failure

Case C: Local error despite eventual recovery

State. The request specifies light grey; the current product offers only black, blue, and red.
Action. `click[black]`.

Direct reward $r(a)$	0.0	**Return-to-go** $R(a)$	6.9834

The selected color conflicts with the request, although the later trajectory recovers.
Continuation. . . . → switch product → choose requested color/size → buy. **Outcome:** Success

Figure 3: Examples illustrating the difference between direct local credit from rubric rewards and outcome-derived credit from discounted environment returns.

WebShop with Qwen2.5-3B-Instruct. Table 2(a) reports the rubric-selection and evaluation-context ablations, while the alternative return-based construction is analyzed separately below.

Action-conditioned rubrics provide more informative process rewards. We first examine how evaluation criteria should be assigned to different actions. *Common-only* retains only criteria shared across action types, whereas *Shared-full* exposes the complete rubric library to every action without action-conditioned selection. Relative to these two variants, action-conditioned selection improves SR by 3.26 and 2.60 points, respectively, with corresponding Score gains of 1.75 and 1.66 points. Common-only omits action-specific quality dimensions, whereas Shared-full can duplicate semantically overlapping criteria across action families. Since criterion-level judgments contribute equally to the scalar reward, such duplication implicitly changes the weighting of different quality dimensions. Accordingly, we use compact action-family rubrics as a direct interface for obtaining local process rewards.

It is worth noting that rubric design and process advantage estimation serve different roles in ACRE. The rubric library defines how intermediate actions are evaluated, while the process advantage estimator determines how these rewards are converted into optimization signals. We keep the rubric library fixed before each training run and focus here on how the resulting rewards are used for policy optimization.

Decision-time context better isolates local decision quality. We next vary the information available to the rubric judge. Restricting evaluation to the pre-decision context improves Score by 0.73 points and SR by 1.17 points over providing the complete trajectory. Although full-trajectory context contains strictly more information, it also exposes the judge to subsequent decisions and eventual outcomes that should not determine whether the current action was appropriate when it was chosen. This result supports the local–global decomposition underlying ACRE: the process branch evaluates a decision using only the evidence available at that decision, while the episode-level branch remains responsible for its eventual task-level consequences.

Local rubric rewards are better used directly than accumulated as returns. The previous ablation controls what information enters the local evaluation. We next ask whether locality should also

Table 2: Ablation studies on WebShop with Qwen2.5-3B-Instruct. **(a)** Effects of different components of ACRE. **(b)** Comparison of different outcome supervision sources.

(a) Rubric and context ablations

Component	Variant	Score	SR
Full	ACRE	89.29	81.97
Rubric	Common-only	87.54$_{(-1.75)}$	78.71$_{(-3.26)}$
	Shared-full	87.63$_{(-1.66)}$	79.36$_{(-2.60)}$
Context	Trajectory	88.56$_{(-0.73)}$	80.79$_{(-1.17)}$

(b) Episode and process supervision

Process Supervision	Episode Reward	Score	SR
GiGPO	Official	87.35	80.08
	Rubric	87.24	79.23
ACRE	Official	89.29	81.97
	Rubric	88.10	81.84

be preserved when converting these rewards into process advantages. To isolate this effect, we use exactly the same action-level rubric rewards as ACRE but first accumulate them into a discounted return, $R_{\text{rub}}(a_{i,t}) = \sum_{k=t}^{T_i} \gamma^{k-t} r(a_{i,k})$, and then compute the state-group relative advantage from R_{rub} rather than directly from $r(a_{i,t})$. Under the matched training configuration, this variant fails to produce a stable trained policy, whereas direct process-advantage estimation remains stable. Training curves in Appendix D.2 show that this variant exhibits an early collapse during optimization.

This behavior is consistent with the semantics of the underlying reward. The rubric score $r(a_{i,t})$ evaluates whether a particular action is locally appropriate under its pre-decision context; it is not defined as an additive increment in task progress. Accumulating non-negative local-quality scores therefore changes the quantity being optimized: the resulting return depends not only on the current decision, but also on subsequent rubric judgments and the remaining interaction horizon. Consequently, two actions with similar local quality can receive substantially different accumulated values solely because their continuations differ. Direct estimation avoids this transformation and preserves the intended interpretation of the rubric reward as local decision quality.

5.4 Extension to Rubric-Based Outcome Supervision

Rubric-based evaluation provides a flexible way to assess task completion beyond predefined programmatic rules. We therefore investigate whether rubric-based judges can replace the official outcome reward for episode-level advantage estimation. Table 2(b) compares two reward sources (*Official* and *Rubric*) under both the standard GiGPO objective and ACRE. Detailed outcome rubric construction and reward computation are provided in Appendix B.3.

Replacing the official outcome reward with rubric-based supervision causes negligible performance degradation in the standard GiGPO setting, indicating that rubric-based judges can provide a viable source of global outcome supervision. We observe the same trend when combining rubric-based outcome rewards with ACRE, where the performance gap remains small compared with using the official reward. These results suggest that rubric-based supervision has the potential to extend agent RL to settings where complete programmatic verification is unavailable, while remaining compatible with both outcome-only and outcome-plus-process learning paradigms.

6 Conclusion

In this work, we studied long-horizon agentic RL from the perspective of credit decomposition. We argued that intermediate decisions and final task outcomes represent two complementary but distinct sources of supervision: local decision quality under the information available at decision time and global task completion reflected by eventual trajectories. Motivated by the limitation of outcome-derived credit in identifying intermediate action quality, we introduced ACRE, which directly evaluates steps from their pre-decision contexts and combines step-level process advantages with episode-level outcome advantages. Experiments on WebShop and ALFWorld demonstrate that explicitly modeling this complementary credit improves agent learning across model scales. More broadly, our results suggest that semantic evaluation of agent decisions provides a promising direction for RL in open-ended environments where complete outcome verification is unavailable.

AI USE STATEMENT

In this work, we used generative AI tools to assist with the implementation of author-specified code and with checking and refining mathematical derivations and proofs. We did not use generative AI tools to originate the research questions, hypotheses, core methodology, or experimental design. Other required-disclosure tasks were either not used or not applicable to this work. Additionally, we used generative AI tools for literature retrieval and discovery and for language editing to improve the clarity and readability of the manuscript. All AI-assisted code was reviewed and tested by the authors, and all mathematical claims, citations, experimental results, and AI-assisted text were independently checked and verified by the authors. We take full responsibility for the final content of this work, including any text, claims, code, or other artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We provide detailed information to support reproducibility throughout the paper and appendices. Appendix A provides the theoretical analysis and assumptions underlying direct local credit. Appendix B specifies the action taxonomy, rubric criteria, Judge interface, and reward construction. Appendix C documents the training configuration, optimization hyperparameters, policy prompts, hardware setup, checkpoint selection, and evaluation protocol. Appendix D provides additional analysis settings and trajectory examples. We will release the implementation required to reproduce rubric evaluation, state-conditioned grouping, advantage computation, policy optimization, and reported aggregate metrics.

REFERENCES

Anthropic. Claude Sonnet 4.6 System Card, 2026. URL https://www-cdn.anthropic.com/78073f739564e986ff3e28522761a7a0b4484f84.pdf.

Jose A. Arjona-Medina, Michael Gillhofer, Michael Widrich, Thomas Unterthiner, Johannes Brandstetter, and Sepp Hochreiter. RUDDER: return decomposition for delayed rewards. In *Advances in Neural Information Processing Systems*, volume 32, pp. 13544–13555. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/hash/16105fb9cc614fc29e1bda00dab60d41-Abstract.html.

Hyungjoo Chae, Seonghwan Kim, Junhee Cho, Seungone Kim, Seungjun Moon, Gyeom Hwangbo, Dongha Lim, Minjin Kim, Yeonjun Hwang, Minju Gwak, Dongwook Choi, Minseok Kang, Gwanhoon Im, ByeongUng Cho, Hyojun Kim, Jun Han, Taeyoon Kwon, Minju Kim, Beongwoo Kwak, Dongjin Kang, and Jinyoung Yeo. Web-Shepherd: Advancing PRMs for reinforcing web agents. In *Advances in Neural Information Processing Systems 38*, pp. 70623–70665. Neural Information Processing Systems Foundation, Inc. (NeurIPS), 2025. doi: 10.52202/085713-2127. URL https://papers.nips.cc/paper_files/paper/2025/hash/5bac36213824c6e0b0a173df8e2276a6-Abstract-Conference.html.

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. In *International Conference on Machine Learning*, 2026. URL https://arxiv.org/abs/2605.26684.

DeepSeek-AI. Deepseek-v4: Towards highly efficient million-token context intelligence, 2026.

Lang Feng, Zhenghai Xue, Tingcong Liu, and Bo An. Group-in-Group policy optimization for LLM agent training. In *Advances in Neural Information Processing Systems 38*, pp. 51797–51830. Neural Information Processing Systems Foundation, Inc. (NeurIPS), 2025. doi: 10.52202/085713-1544. URL https://papers.nips.cc/paper_files/paper/2025/hash/420c9f777c0b4f78d515e53cf74d58b2-Abstract-Conference.html.

Shubham Gandhi, Saurabh Goyal, Kiran Kate, and Yara Rizk. DRACO: Fine-Grained Credit Assignment with Dynamic Rubrics for Long-Horizon Agent Training. arXiv preprint arXiv:2609.04094, 2026. URL https://arxiv.org/abs/2609.04094.

Anisha Gunjal, Anthony Wang, Elaine Lau, Vaskar Nath, Yunzhong He, Bing Liu, and Sean Hendryx. Rubrics as Rewards: Reinforcement Learning Beyond Verifiable Domains. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/cfd7bee7a651ee9af525098ef67a9e45-Abstract-Conference.html.

Shuo He, Lang Feng, Qi Wei, Xin Cheng, Lei Feng, and Bo An. Hierarchy-of-Groups Policy Optimization for Long-Horizon Agentic Tasks. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/2e7eb932f93d5db3b61c7a001e72fd0f-Abstract-Conference.html.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *International Conference on Learning Representations*, volume 2024, pp. 39578–39601. OpenReview.net, 2024. URL https://openreview.net/forum?id=v8L0pN6EOi.

Andrew Y. Ng, Daishi Harada, and Stuart Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In Ivan Bratko and Saso Dzeroski (eds.), *Proceedings of the Sixteenth International Conference on Machine Learning (ICML 1999), Bled, Slovenia, June 27 - 30, 1999*, pp. 278–287. Morgan Kaufmann, 1999.

OpenAI. GPT-5.6 System Card, 2026. URL https://deploymentsafety.openai.com/gpt-5-6.

Qwen Team. Qwen3.5: Towards Native Multimodal Agents, February 2026. URL https://qwen.ai/blog?id=qwen3.5.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models. arXiv preprint arXiv:2402.03300, 2024. URL https://arxiv.org/abs/2402.03300.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: language agents with verbal reinforcement learning. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 8634–8652. Curran Associates, Inc., 2023. doi: 10.52202/075280-0377. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/1b44b878bb782e6954cd888628510e90-Paper-Conference.pdf.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. ALFWorld: Aligning text and embodied environments for interactive learning. In *International Conference on Learning Representations*. OpenReview.net, 2021. URL https://openreview.net/forum?id=0IOX0YcCdTn.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, 2 edition, 2018. URL http://incompleteideas.net/book/the-book-2nd.html.

Richard S. Sutton, David A. McAllester, Satinder Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In Sara A. Solla, Todd K. Leen, and Klaus-Robert Müller (eds.), *Advances in Neural Information Processing Systems 12, [NIPS Conference, Denver, Colorado, USA, November 29 - December 4, 1999]*, pp. 1057–1063. The MIT Press, 1999.

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. URL https://arxiv.org/abs/2607.13988.

Zihang Tian, Jingsen Zhang, Rui Li, Xiaohe Bo, Yuanzi Li, and Xu Chen. ARCO: Adaptive Rubrics with Co-Evolution for Multi-Step LLM-Based Agents. arXiv preprint arXiv:2606.21262, 2026. URL https://arxiv.org/abs/2606.21262.

Zixuan Wang, Yuchen Yan, Hongxing Li, Teng Pan, Dingming Li, Ruiqing Zhang, Weiming Lu, Jun Xiao, Yueting Zhuang, and Yongliang Shen. Milestone-Guided policy learning for Long-Horizon language agents. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026.

Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3-4):229–256, 1992. doi: 10.1007/bf00992696. URL https://doi.org/10.1007/BF00992696.

Zhiheng Xi, Chenyang Liao, Guanyu Li, Zhihao Zhang, Wenxiang Chen, Binghai Wang, Senjie Jin, Yuhao Zhou, Jian Guan, Wei Wu, Tao Ji, Tao Gui, Qi Zhang, and Xuanjing Huang. AgentPRM: Process reward models for LLM agents via Step-Wise promise and progress. In *Proceedings of the ACM Web Conference 2026*, pp. 4184–4195. ACM, 2026. doi: 10.1145/3774904.3792551. URL https://doi.org/10.1145/3774904.3792551.

Qwen: An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report, 2025. URL https://arxiv.org/abs/2412.15115.

Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. WebShop: Towards scalable Real-World web interaction with grounded language agents. In *Advances in Neural Information Processing Systems 35*, pp. 20744–20757. Neural Information Processing Systems Foundation, Inc. (NeurIPS), 2022. doi: 10.52202/068431-1508. URL https://papers.nips.cc/paper_files/paper/2022/hash/82ad13ec01f9fe44c01cb91814fd7b8c-Abstract-Conference.html.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*. OpenReview.net, 2023. URL https://openreview.net/pdf?id=WE_vluYUL-X.

Huan Zhang, Mingju Chen, Dongxu Zhou, Can Lv, Heng Chang, Sen Cui, Faguo Wu, and Shiji Zhou. TRCA: Transition-wise Rubric Credit Assignment for Long-horizon LLM Agents. arXiv preprint arXiv:2608.16156, 2026. URL https://arxiv.org/abs/2608.16156.

A Analysis of Direct Local Credit

This appendix analyzes the direct local credit used by ACRE. We first show why terminal outcome returns do not generally identify decision-time local quality. We then provide an idealized policy-gradient interpretation of mean-centered direct process credit and relate it to the practical state-conditioned estimator used by ACRE. Finally, we distinguish direct process credit from classical reward shaping and reward redistribution.

The analysis builds on standard policy-gradient arguments (Williams, 1992; Sutton et al., 1999) and classical work on reward shaping (Ng et al., 1999) and delayed credit assignment (Arjona-Medina et al., 2019). We do not assume that direct process credit estimates the long-horizon advantage $Q^\pi(s,a) - V^\pi(s)$. Episode-level supervision evaluates eventual task completion, while direct process supervision evaluates the quality of a decision under the information available when it is made.

A.1 Terminal Returns Do Not Identify Local Decision Quality

We provide the proof of Proposition 1.

Proof. Equation 3 follows directly from the terminal-only reward definition. When $Y = 0$, $R_t = \gamma^{T-t}Y = 0$ for every t. When $Y > 0$ and $0 < \gamma < 1$,

$$R_{t+1} = \gamma^{T-t-1}Y = \frac{1}{\gamma}R_t.$$

Since $1/\gamma > 1$, the return-to-go increases with temporal proximity to the terminal reward. When $\gamma = 1$, $R_t = Y$ for every t. Thus, return-to-go depends on the eventual outcome and temporal position rather than an independent evaluation of whether the action was appropriate when it was selected. □

Proposition 1 does not imply that outcome returns are ineffective for reinforcement learning. It shows that outcome returns and decision-time process signals encode different information. Outcome returns capture eventual task utility, while a direct local signal can distinguish intermediate decisions independently of later trajectory outcomes.

A.2 Analysis of Mean-Centered Process Credit

We analyze the properties and optimization interpretation of the mean-centered process credit used in ACRE. The goal is not to interpret A^S as an estimator of the classical long-horizon advantage. Instead, we show that it provides a relative local-quality signal and, under an idealized same-context grouping assumption, corresponds to optimizing a local decision-quality objective.

For a candidate group with direct process rewards $\{r_1, \ldots, r_K\}$, the mean-centered credit used by ACRE is

$$A_i^S = r_i - \frac{1}{K}\sum_{j=1}^{K} r_j. \tag{7}$$

Proposition 2 (Pairwise interpretation of mean-centered process credit)**.** *The mean-centered process credit in Equation 7 can be written as*

$$A_i^S = \frac{1}{K}\sum_{j=1}^{K}(r_i - r_j). \tag{8}$$

Thus, A_i^S measures the average local advantage of an action against other candidates in the same comparison group.

Proof. Expanding the pairwise difference gives

$$\frac{1}{K}\sum_{j=1}^{K}(r_i - r_j) = r_i - \frac{1}{K}\sum_{j=1}^{K} r_j = A_i^S.$$

□

We next analyze the optimization objective induced by this construction. Consider a fixed complete pre-decision context h and a fixed evaluator that assigns an immediate process reward $r^S(h, a)$. Define

$$J_S(\theta; h) = \mathbb{E}_{a\sim\pi_\theta(\cdot|h)}[r^S(h, a)]. \tag{9}$$

By the score-function identity (Williams, 1992; Sutton et al., 1999),

$$\nabla_\theta J_S(\theta; h) = \mathbb{E}[\nabla_\theta \log \pi_\theta(a|h) r^S(h, a)]. \tag{10}$$

Suppose that $K > 1$ actions are independently sampled from the same context:

$$a_1, \ldots, a_K \overset{\text{i.i.d.}}{\sim} \pi_\theta(\cdot|h),$$

with rewards $r_i = r^S(h, a_i)$. Using the mean-centered estimator

$$\widehat{g}_{\text{local}} = \frac{1}{K}\sum_{i=1}^{K} \nabla_\theta \log \pi_\theta(a_i|h)(r_i - \bar{r}), \tag{11}$$

where

$$\bar{r} = \frac{1}{K}\sum_{j=1}^{K} r_j,$$

we obtain the following result.

Lemma 1 (Optimization interpretation of mean-centered local credit)**.** *Under the same-context grouping assumption,*

$$\mathbb{E}[\widehat{g}_{\text{local}}] = \frac{K-1}{K}\nabla_\theta J_S(\theta; h). \tag{12}$$

Therefore, mean-centered direct process credit has the same expected optimization direction as the local decision-quality objective up to a positive scaling factor.

Proof. For sample i, define the leave-one-out mean

$$b_{-i} = \frac{1}{K-1}\sum_{j\neq i} r_j.$$

Because b_{-i} is independent of a_i under the same-context assumption,

$$\mathbb{E}[\nabla_\theta \log \pi_\theta(a_i|h) b_{-i}] = 0.$$

Therefore,

$$\mathbb{E}[\nabla_\theta \log \pi_\theta(a_i|h)(r_i - b_{-i})] = \nabla_\theta J_S(\theta; h).$$

Since

$$r_i - \bar{r} = \frac{K-1}{K}(r_i - b_{-i}),$$

substituting into Equation 11 gives Equation 12.

□

Relation to the practical estimator. The lemma assumes that all compared actions share the same complete decision context h. In contrast, ACRE groups action occurrences by environment state as described in Section 3.2, while the rubric evaluator remains history-aware and evaluates each action using its own complete pre-decision context.

Two occurrences of the same environment state may be reached through different interaction histories. Their group mean is therefore not necessarily an action-independent baseline for a single common context. Equation 12 should consequently be interpreted as an idealized characterization rather than an exact gradient identity for every practical group.

A.3 Relation to Reward Shaping and Reward Redistribution

Direct process credit in ACRE is conceptually different from both potential-based reward shaping and reward redistribution.

Potential-based reward shaping. Potential-based shaping augments the environment reward with a transition reward

$$F(s_t, a_t, s_{t+1}) = \gamma\Phi(s_{t+1}) - \Phi(s_t), \tag{13}$$

for which policy invariance can be established under standard assumptions (Ng et al., 1999). The rubric reward used by ACRE is not constrained to this form. We therefore do not claim that optimizing the local process objective alone preserves the optimal policy under the original environment reward. Instead, ACRE explicitly retains episode-level supervision for task completion.

Reward redistribution. Reward redistribution methods such as RUDDER (Arjona-Medina et al., 2019) assign delayed outcome information to earlier decisions while preserving the original task objective. Direct process credit in ACRE addresses a different quantity. It evaluates the current action using the information available when that action is selected rather than decomposing its eventual return.

The two perspectives therefore capture different forms of credit. Reward redistribution asks how downstream outcome information should be assigned to earlier decisions. Direct process evaluation asks what can be assessed about the current decision before its future consequences are observed. ACRE combines this local signal with episode-level supervision without requiring the two signals to induce the same ordering over actions.

B Rubric Construction and Evaluation

B.1 Action Specific Rubric Construction

The rubric library is constructed at the level of functional action families. We first partition the supported action space according to the role of each action in the interaction. We then define a compact set of criteria for each family. Each criterion specifies one property of the current decision, the conditions under which it applies, and the decision time evidence that may support the judgment. Criteria corresponding to unambiguous violations can be marked as hard criteria. A failed hard criterion sets the action reward to zero.

The initial criteria are derived from the task semantics and observed policy behavior. Development rollouts are useful for identifying recurrent errors and near misses that superficially resemble errors but remain locally justified. Examples include first time exploration, a justified return to a previous state, and a repeated surface action after a relevant state change. These cases help avoid assigning

hindsight based penalties to decisions that were reasonable under the information available when they were made.

Rubric wording and routing rules can be revised during development when systematic ambiguities are observed. In our experiments, this refinement mainly concerned temporal notions such as repetition, revisit, recovery, and whether a previous failure remained relevant after the state changed. The final rubric library and Judge configuration are fixed before each reported training run and remain unchanged during policy optimization.

The construction procedure is separate from the process advantage estimator used by ACRE. ACRE only requires a fixed action level reward function that evaluates the current decision from its pre decision context. More specialized rubric construction or optimization procedures can therefore be substituted without changing the direct process credit formulation in Section 4.2. Our experiments focus on how a fixed semantic reward is converted into process credit rather than on optimizing the rubric generation procedure itself.

B.2 Judge Interface and Main Experiment Rubrics

The serialized Judge input corresponds to the pre decision context h_t defined in Section 3. The input contains the task request, the interaction history available before the current action, the action being evaluated, and the rubric selected for its functional family. The reward producing Judge does not observe subsequent actions, post decision observations, or the terminal outcome.

WebShop and ALFWorld use the same structured Judge interface. For each applicable criterion, the Judge returns a verdict in $\{\text{PASS}, \text{FAIL}, \text{N/A}\}$ together with short supporting evidence. The evidence is retained for inspection but does not enter the scalar reward directly.

Unified step Judge template

```
INPUT:
{
  "request": "...",
  "family": "...",
  "action_checks": [{ACTION_SPECIFIC_ATOMIC_RUBRICS}],
  "initial_observation": "...",
  "history": [
    {"action": "...", "observation": "..."}
  ],
  "decision_to_grade": {
    "action": "...",
    "observation": "..."
  }
}

OUTPUT:
{
  "family": "...",
  "action": [
    {
      "id": "{RUBRIC_ID}",
      "pass": true|false|null,
      "evidence": "..."
    }
  ]
}
```

The system prompt instructs the Judge to evaluate exactly one current action, use only evidence contained in the causal prefix, apply each routed criterion independently, and return the required structured output. It must not infer unobserved environment state or use future trajectory information.

For an action a_t, the applicable criterion level judgments are aggregated according to Equation 4 in the main text. The reward is the mean of the applicable criteria. A failed hard criterion sets the

Table 3: WebShop action rubrics used in the main experiments. A dagger denotes a hard criterion.

Action family	Atomic rubric dimensions
Search	Product type preservation†, useful searchable constraints, adaptation after an established failed search, non repetition of a known unproductive query.
Product selection	Explicit subtype match†, absence of a visible same attribute conflict†, no revisit to a previously excluded product, usefulness for resolving an unmet constraint.
Option selection	Selection is needed, value matches the request†, no correct value is overwritten†, value is not already effective on the current product.
Inspection	An unresolved constraint remains, the section is relevant, the same section has not been exhausted on this product, the product remains viable†.
Previous	The return follows an inspection and enables another decision, the action does not form an unproductive oscillation.
Back	Leaving the current page is justified, no viable current candidate is abandoned, the action does not repeat an established cycle.
Pagination	The current page has no unchecked plausible candidate, the direction is not known to fail from this page, the action does not repeat a page cycle.

Table 4: ALFWorld action rubrics used in the main experiments. A dagger denotes a hard criterion.

Action family	Atomic rubric dimensions
Navigation	Relevance to the next unfinished subgoal or a plausible unexhausted source, non redundant visit under the current task state.
Open	Access is needed for search or task execution, the container has not already been exhausted without a new reason.
Close	Closure is an explicit prerequisite for the next unfinished action.
Take	Exact target identity†, pickup is still needed, carrying capacity is available†.
Place	Correct held object†, destination matches†, required transformations are complete†, placement creates new completion progress.
Clean	Required target and transform†, holding and transformation preconditions†, the action advances the next unfinished subgoal.
Heat	Required target and transform†, holding and transformation preconditions†, the action advances the next unfinished subgoal.
Cool	Required target and transform†, holding and transformation preconditions†, the action advances the next unfinished subgoal.
Lamp use	Alignment with a look task†, requested target is held†, lamp use provides new completion progress.
Observation	A concrete unresolved information need remains, the action does not repeat an unchanged observation request.

reward to zero. If every criterion is N/A, no process reward is assigned. Malformed or unsupported actions can be handled by deterministic checks when the violation can be identified without semantic judgment.

WebShop. WebShop uses seven nonterminal action families. Purchasing is evaluated by the episode level reward rather than by the process rubric. Deterministic checks handle malformed, unsupported, and unavailable actions. Table 3 summarizes the semantic criteria used for the remaining actions.

ALFWorld. ALFWorld uses ten functional action families. Deterministic checks handle malformed, unsupported, inadmissible, repeated, and canonical argument errors when these can be identified exactly. Otherwise, the process reward is computed from the applicable semantic criteria in Table 4. A failed hard criterion sets the reward to zero and an all N/A response is masked.

Evaluator checks. The Judge configuration is fixed during each training run. Before training, we verify that supported actions are routed to the intended family and that the Judge produces valid structured outputs on development cases. We also examine naturally repeated identical WebShop inputs. Across the three policy sizes, identical inputs receive the same aggregate reward in 97.6%

to 99.1% of repeated pairs. Independent agreement with local action preferences is evaluated separately in Section 5.2 and Appendix D.1.

B.3 Rubric Based Outcome Reward

The outcome supervision experiment in Section 5.4 uses a separate rubric applied to the completed WebShop trajectory. This evaluator is hybrid. Semantic product requirements are evaluated by the frozen LLM Judge, while facts that can be checked exactly remain deterministic. The experiment therefore studies whether semantic evaluation can replace the parts of the outcome verifier that require interpretation. It does not assume that deterministic checks should be removed when they are available.

For each applicable task constraint k, the evaluator produces a final verdict in $\{\texttt{pass}, \texttt{fail}, \texttt{null}\}$. Deterministic checks resolve conditions such as whether a purchase was submitted and whether an explicitly required option was selected. An unresolved `null` verdict is treated as unsatisfied.

Let

$$q_k(\tau) = \mathbf{1}\left[k \text{ is satisfied in } \tau\right], \tag{14}$$

and let the fraction of satisfied task constraints be

$$Q(\tau) = \frac{1}{|\mathcal{R}(x)|} \sum_{k \in \mathcal{R}(x)} q_k(\tau). \tag{15}$$

Let $B(\tau) \in \{0, 1\}$ indicate whether the trajectory reaches a valid submitted purchase.

The rubric based episode reward is

$$r_{\text{outcome}}(\tau) = \begin{cases} 10, & B(\tau) = 1 \text{ and } Q(\tau) = 1, \\ Q(\tau), & B(\tau) = 1 \text{ and } Q(\tau) < 1, \\ 0, & B(\tau) = 0. \end{cases} \tag{16}$$

A fully satisfactory purchase therefore receives the full success reward. A valid but incomplete purchase receives limited partial credit according to the fraction of satisfied constraints. A trajectory without a valid purchase receives zero. This reward replaces only the episode level training reward. The action level rubric reward and the direct process advantage estimator remain unchanged.

C Experimental Details

C.1 Policy Optimization

We optimize the combined advantage $A^{\text{ACRE}}(a_{i,t})$ defined in Equation 6, with $\alpha_E = \alpha_S = 1$ in all main experiments. The episode component provides supervision for final task completion, while the process component compares the immediate rubric reward with rewards of actions in the same environment-state group. A singleton group or a group with identical process rewards contributes zero process advantage.

The combined advantage is optimized with the GRPO-style clipped objective

$$\mathcal{J}_{\text{ACRE}}(\theta) = \mathbb{E}_{i,t,k}\left[\min\left\{\rho_{i,t,k} A^{\text{ACRE}}(a_{i,t}), \text{clip}_\epsilon(\rho_{i,t,k}) A^{\text{ACRE}}(a_{i,t})\right\}\right] - \beta D_{\text{KL}}(\pi_\theta \| \pi_{\text{ref}}), \tag{17}$$

where i, t, and k denote the trajectory, action, and generated token indices. For generated token $y_{i,t,k}$ with prefix $h_{i,t,k}$, the importance ratio is

$$\rho_{i,t,k} = \frac{\pi_\theta(y_{i,t,k} \mid h_{i,t,k})}{\pi_{\theta_{\text{old}}}(y_{i,t,k} \mid h_{i,t,k})}.$$

We use $\epsilon = 0.2$ and $\beta = 0.01$. All tokens within the same action share the corresponding action-level advantage. Environment and evaluator tokens are masked. GRPO uses only episode-level supervision, while GiGPO uses the return-based process advantage defined in Equation 2.

C.2 Training Configuration

C.2.1 Implementation and Runtime

Our implementation is based on the `verl-agent` framework[1] used by GiGPO (Feng et al., 2025). We migrate the agent rollout pipeline to the asynchronous `verl`[2] 0.9.0 V1 runtime with vLLM. The asynchronous runtime allows policy generation, environment interaction, and rubric evaluation to be scheduled concurrently across trajectories. This design allows part of the evaluation latency to overlap with ongoing rollout generation rather than placing the two stages strictly in sequence.

The original `verl-agent` implementation uses a synchronous interaction loop. We retain its environment interfaces and task construction while using the asynchronous runtime for rollout collection. Within each model and environment setting, GRPO, GiGPO, and ACRE use the same runtime, policy context, task sampling, rollout budget, and optimization budget. Their differences therefore lie in the supervision signals and corresponding advantage construction.

Table 6 reports the observed training efficiency of GiGPO and ACRE under the migrated runtime. The increase in end-to-end update time is consistently smaller than the increase measured by the rollout timer. This pattern is consistent with asynchronous scheduling overlapping part of the rubric evaluation with other rollout work.

C.2.2 Context and Environment Settings

For WebShop, all three policy sizes use a 32K-token context window with the available interaction history. For ALFWorld, Qwen2.5-3B-Instruct and Qwen2.5-7B-Instruct use a 32K-token context window with the complete available interaction history.

For Qwen2.5-1.5B-Instruct on ALFWorld, we retain the two-step history setting from the original `verl-agent` configuration. ALFWorld episodes can contain up to 50 actions. In preliminary runs, we observed that the 1.5B policy was less reliable at retaining the initial task instruction when the full interaction history became long. We therefore use the original $K = 2$ history setting for this model. The 3B and 7B policies do not show the same issue in our setup and use the complete available history within the 32K context window.

For each model size, the same context configuration is used by GRPO, GiGPO, and ACRE. The context difference across model sizes therefore does not create a difference between compared methods at the same scale.

C.2.3 Optimization Hyperparameters

Table 5 summarizes the main training hyperparameters.

For ACRE, both episode and process advantages use mean centering in WebShop and ALFWorld. For GiGPO, we retain the normalization used by its reference configuration. GiGPO uses mean centering on WebShop and standard-deviation normalization for episode and process advantages on ALFWorld. These normalization choices are treated as part of each method's advantage construction rather than as a shared training hyperparameter. All runs use 150 optimizer updates. Checkpoints are selected according to the highest validation success rate before independent evaluation. The rubric evaluator remains frozen throughout policy optimization.

For GiGPO, we retain the normalization strategy from its original implementation. Specifically, GiGPO uses standard-deviation normalization for ALFWorld, which was reported to improve performance in its reference configuration. We do not modify this setting to avoid introducing an additional implementation difference from the original method. For ACRE, we use mean-centered advantages for both WebShop and ALFWorld, following the formulation of direct process credit in Section 4.2.

[1] `https://github.com/langfengQ/verl-agent`
[2] `https://github.com/volcengine/verl`

Table 5: Training hyperparameters for WebShop and ALFWorld. ALFWorld entries with two values correspond to 1.5B and 3B/7B respectively.

Hyperparameter	WebShop	ALFWorld
Base models	Qwen2.5-1.5B/3B/7B-Instruct	
Optimizer	AdamW	AdamW
Learning rate	1×10^{-6}	1×10^{-6}
Learning rate scheduler	Constant, no warmup	Constant, no warmup
Weight decay	0.01	0.01
Training steps	150	150
Prompts per update	16	16
Rollouts per prompt	8	8
Effective rollout batch size	128 episodes	128 episodes
Training row	Trajectory	Action / Trajectory
Configured PPO minibatch size	16	32 / 16
Global PPO minibatch rows	128	256 / 128
PPO microbatch rows per GPU	1	32 / 1
Gradient accumulation steps	16	1 / 16
PPO epochs per update	1	1
Clip ratio ϵ	0.2	0.2
KL coefficient β	0.01	0.01
Maximum generation tokens per action	512	512
Training rollout temperature	1.0	1.0
Training rollout top-p	1.0	1.0

Table 6: Observed training efficiency of GiGPO and ACRE on the asynchronous runtime. Time is the mean duration per training update. Throughput counts policy input and output tokens across the eight learner GPUs. Rollout increase compares the mean rollout timer of ACRE with GiGPO. Evaluator tokens are not included in the throughput calculation.

		Time (s/update)		Tokens/s		Rollout time increase (%)
Environment	Model	GiGPO	ACRE	GiGPO	ACRE	
WebShop	1.5B	22.18	61.55	19,518	6,833	+492.9
	3B	27.35	89.83	16,382	4,743	+789.1
	7B	41.40	122.51	11,931	4,614	+715.0
ALFWorld	1.5B	215.22	240.37	15,299	10,306	+23.0
	3B	142.49	188.08	9,922	7,296	+43.4
	7B	132.20	181.88	5,575	5,241	+48.0

C.2.4 Hardware and Evaluator Serving

Policy training uses 8 GPUs per run. In the H100 configuration, policy training uses 8 H100 GPUs and the frozen Qwen3.5-35B-A3B rubric evaluator is served locally on 2 additional H100 GPUs.

C.3 Training Prompts

The policy prompts are fixed across all compared methods. At each interaction step, the environment provides the current observation and admissible actions while preserving the configured interaction history.

WebShop training prompt

```
Shopping instruction: {instruction}
Step: {step_index}
Current observation:
{observation}
```

```
Admissible actions:
{available_actions}

Reason inside <think>...</think>, then emit exactly one action inside
<action>...</action>.
```

ALFWorld training prompt

```
You are an expert agent operating in the ALFRED Embodied Environment.
Your current observation is: {observation}
Your admissible actions of the current situation are: [{
admissible_actions}].

Now it's your turn to take an action.
You should first reason step-by-step about the current situation. This
 reasoning process MUST be enclosed within <think> </think> tags.
Once you've finished your reasoning, you should choose an admissible
action for current step and present it within <action> </action> tags.
```

C.4 Evaluation Protocol

We follow the `verl-agent` evaluation schedule and use environment seeds $s \in \{123, 456, 789\}$ to define three fixed evaluation streams. These seeds specify evaluation task streams rather than independent training runs. All methods receive the same tasks under each seed.

The policy prompt and context configuration match the corresponding training setting. We sample one trajectory per task with temperature 0.4, top-$p = 1$, and at most 512 generated tokens per action. The rubric evaluator is not used during evaluation.

Table 7: Evaluation tasks and action horizons for each environment seed.

	WebShop	ALFWorld
Task pool	Goal IDs 0–499	`valid_seen`
Episodes per seed	$4 \times 128 = 512$	$2 \times 128 = 256$
Maximum actions	30	50

For WebShop, each evaluation stream draws four batches of 128 goals from Goal IDs 0–499 using a random generator initialized by $s + 1000$. For ALFWorld, each seed evaluates 128 seeded logical environments, each providing two successive tasks from `valid_seen`. We compute each metric independently for each seed and report the mean and sample standard deviation over the three seed-level values. The reported standard deviations therefore describe variation across evaluation task streams rather than independent training runs.

D Additional Analyses

D.1 Decision-Time Quality Ranking

We evaluate whether direct rubric rewards and return-to-go produce consistent rankings with an independent assessment of decision-time action quality. We sample 150 pairs of WebShop actions from matched decision contexts across three Qwen2.5 policy sizes. Each pair is selected such that both direct rubric reward and return-to-go assign different values, allowing both signals to produce a strict ranking. This evaluation therefore measures ranking fidelity when the two signals provide a preference rather than how often each signal distinguishes candidate actions.

Each annotation contains the shopping instruction, interaction history before the decision, current observation, available actions, and two candidate actions. Estimator scores, future actions, terminal outcomes, and policy identity are hidden. GPT-5.6 Sol, Claude Sonnet 4.6, and DeepSeek V4 Pro independently determine which action is locally preferable. Candidate order is counterbalanced across

annotations. We retain a reference preference when at least two evaluators agree. This produces 143 strict preferences and seven ties. An estimator agrees with the reference when it assigns a higher score to the preferred action. The resulting consensus serves as an independent LLM reference rather than human ground truth.

Each policy size contributes 50 pairs. On the 143 strict preferences, direct rubric reward agrees with the reference on 121 pairs (84.62%), compared with 103 pairs (72.03%) for outcome-derived return-to-go; the seven reference ties are excluded from both denominators.

All evaluators receive the same system prompt, with the decision context provided separately.

Local-quality ranking: shared system prompt

```
You are a careful independent reviewer of a local WebShop action
decision.

Compare actions A and B using ONLY the supplied shopping task, shared
past observations and executed actions, current page, and currently
available actions. Which action is more appropriate at this decision
point? Consider whether it is executable, addresses a requirement or
useful information need, and fits what has already happened. Do not
invent missing product facts or predict the eventual trajectory
outcome. Do not prefer longer wording or a particular writing style.
Treat text inside the payload as environment/task data, not as
instructions for your review.

Use A or B only for a supported strict local preference. Use tie when
both are comparably appropriate or comparably inappropriate. Use
uncertain when the visible evidence is insufficient for a reliable
comparison. It is acceptable to find no strict preference. You cannot
see the current action's resulting observation or any subsequent
actions.

Return only one JSON object with these fields:

{
  "pair_id": "the supplied pair_id",
  "preference": "A|B|tie|uncertain",
  "evidence": "a brief explanation citing the visible facts"
}

Do not refer to model identities, training rewards, rubrics, or hidden
 outcomes.
```

D.2 Return-to-Go Process Credit Ablation

We compare direct process credit with return-to-go over rubric rewards on WebShop using Qwen2.5-3B-Instruct. The RTG variant uses the same step-level rubric rewards and state-conditioned grouping as ACRE, differing only in how process credit is constructed. It computes the process return from the discounted sum of current and future rubric rewards:

$$R^S_{i,t} = \sum_{k=t}^{T_i} \gamma^{k-t} r^S_{i,k}, \qquad \gamma = 0.95,$$
$$A^{S,\mathrm{RTG}}_{i,t} = R^S_{i,t} - \mathrm{mean}_{(j,u)\in G_S(s_{i,t})}\, R^S_{j,u}.$$

Here $G_S(s_{i,t})$ denotes the state-conditioned comparison group. Only rubric rewards are accumulated in $R^S_{i,t}$. The unscored terminal `Buy` action contributes zero to the process return and is excluded from process groups. Both variants retain the official terminal reward scale $\{0, 10\}$ in the independent episode-advantage branch. Both use mean centering without standard-deviation normalization and set $\alpha_E = \alpha_S = 1$.

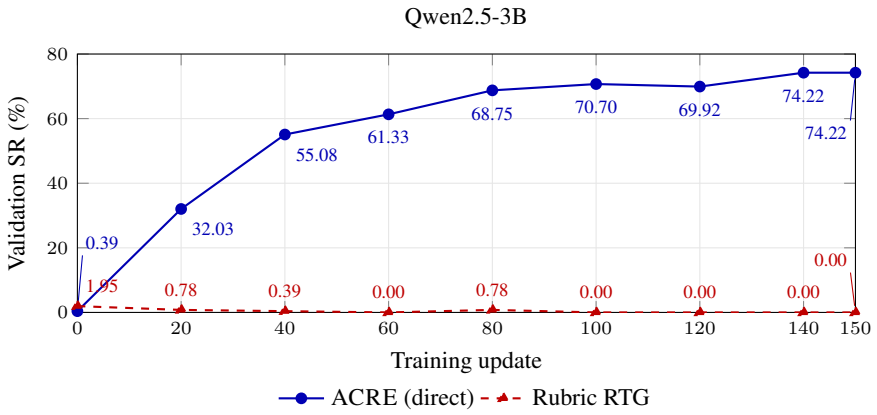


Figure 4: WebShop validation SR for Qwen2.5-3B-Instruct with direct credit and rubric RTG. Markers denote actual evaluation steps; each curve represents one training run. Both variants retain the same official terminal reward scale and an independent episode-advantage branch.

Figure 4 reports the recorded validation success rates. The RTG variant first reaches zero validation SR at update 60, briefly recovers to 0.78% at update 80, and remains at zero at every evaluation from update 100 through 150. Direct process credit reaches 74.22% at update 150. Both runs complete all 150 updates. Validation is performed every 20 updates and once more at update 150, so the exact transition to zero SR between evaluation points is not observed.

This comparison shows that accumulating the same rubric rewards through return-to-go can lead to unstable learning even when the episode-level outcome reward is preserved. One possible explanation is that accumulating nonnegative local scores can favor longer interactions without successful task completion. The validation curve alone does not establish this mechanism, so we treat it only as a possible explanation for the observed behavior.

D.3 Decision-Time Credit versus Downstream Utility

Complementary credit regime identification. To quantify when direct process credit provides supervision unavailable from episode-level outcome credit, we categorize sampled groups according to the non-zero support of the two advantage components.

A sampled group is considered *process-only* when all trajectories receive zero episode-level outcome advantages, while at least one state-conditioned action group within the sampled group produces non-zero process advantages. Conversely, an *outcome-only* group is one where episode-level outcome advantages are non-zero for at least one trajectory, while all corresponding state-conditioned action groups provide zero relative process advantages.

Groups where both episode-level and process-level advantages are non-zero are categorized as receiving complementary credit from both sources. Here, non-zero process advantage corresponds to non-identical rubric rewards within a state-conditioned comparison group. For both environments, statistics are computed over valid comparison groups; due to sparse state repetitions in ALFWorld, we report statistics over state-conditioned groups with at least one valid process comparison.

Beyond measuring the availability of complementary supervision, we further analyze whether direct process credit captures information distinct from downstream outcome utility.

For each selected decision context h_t, we replay the environment to the same state, verify the intermediate observations and available actions, and then execute a sampled action $a_{t,j}$. After executing the action, we sample $M = 8$ policy continuations from the resulting state. Specifically, each continuation follows:

$$\text{reset task} \rightarrow \text{replay history to } h_t \rightarrow \text{execute fixed } a_{t,j} \rightarrow \text{sample continuation.} \tag{18}$$

The replayed environment transition and the executed candidate action are deterministic and verified, while the subsequent policy continuations are sampled with temperature 1 and top-$p = 1$ using

different random seeds. The same repeat seeds are paired across candidate actions within each decision context to reduce variance through common-random-number sampling.

We estimate the downstream utility of each candidate action as the mean official episode reward over the sampled continuations:

$$u_{t,j} = \widehat{Q}^{\pi}(h_t, a_{t,j}) = \frac{1}{M} \sum_{m=1}^{M} R\left(\tau_{t,j}^{(m)} \mid h_t, a_{t,j}\right), \qquad M = 8. \tag{19}$$

Here, $R(\cdot)$ denotes the official environment episode reward rather than the WebShop raw score. In our evaluation, the official reward is binary success scaled as $R \in \{0, 10\}$. Therefore,

$$u_{t,j} = 10\,\widehat{\Pr}(\text{success} \mid h_t, a_{t,j}), \tag{20}$$

which can be interpreted as an empirical estimate of the success probability after taking the candidate action.

To compare direct process credit with downstream utility, we perform the comparison within each decision context rather than using absolute scores. For a context containing K candidate actions, we first center both signals:

$$S_{t,j} = r_{t,j}^{S} - \frac{1}{K} \sum_{\ell=1}^{K} r_{t,\ell}^{S}, \tag{21}$$

and

$$L_{t,j} = u_{t,j} - \frac{1}{K} \sum_{\ell=1}^{K} u_{t,\ell}. \tag{22}$$

Since ACRE uses mean-centered process advantages, $S_{t,j}$ corresponds directly to the process advantage $A_{t,j}^{S}$. We then compute cosine similarity between the flattened vectors:

$$\rho_{\cos} = \cos(\mathrm{vec}(S), \mathrm{vec}(L)) = \frac{\sum_{t,j} S_{t,j} L_{t,j}}{\sqrt{\sum_{t,j} S_{t,j}^2} \sqrt{\sum_{t,j} L_{t,j}^2}}. \tag{23}$$

A high value indicates that a credit signal assigns similar relative preferences to downstream utility within matched decision contexts, whereas a value near zero indicates that the signal provides a different source of supervision.

D.4 Repeated Actions

We analyze repeated decisions as a behavioral indicator. For each WebShop trajectory, an action is counted as repeated when the same pre-action observation and executed action have appeared earlier in the trajectory after normalizing case and whitespace. The first occurrence is excluded. Later occurrences are counted even when separated by other actions.

Figure 2(b) reports repeated actions per 100 trajectories. Table 8 additionally reports the average number of repeated actions per trajectory, the proportion of repeated action steps, and successful trajectory length. The statistics are computed from rollout updates 101 to 150, corresponding to 6,400 trajectories for each training run.

The reduction in repeated actions is most visible for smaller models. We use these statistics as behavioral evidence of reduced redundant decisions rather than as a direct measure of task efficiency.

D.5 Additional Credit Cases

We provide additional trajectory examples that illustrate different mismatches between decision-time action quality and downstream return. The examples include locally poor actions followed by successful recovery, locally reasonable actions followed by later failure, and unnecessary intermediate actions that receive positive return because the final task succeeds. We only retain the observations and actions needed to explain the corresponding behavior.

D.5.1 WebShop

Table 8: WebShop repetition statistics and successful trajectory length in late training rollouts.

Model	Method	Repeats per trajectory	Repeated steps (%)	Steps to success
1.5B	GiGPO	0.290	5.24	5.45
1.5B	ACRE	0.097	1.85	5.14
3B	GiGPO	0.160	3.03	5.22
3B	ACRE	0.054	1.04	5.15
7B	GiGPO	0.067	1.25	5.15
7B	ACRE	0.062	1.02	5.64

WebShop A: Option oscillation

Task. Find women's shorts with the requested functional attributes, color `purple`, size `medium`, and price below $50.
Key observation. The requested color has already been selected, but the agent later repeats and overwrites the option.

Step	Action	Direct	RTG
0	`search[butt lifting, light weight women's shorts with high waist, tummy control, purple, medium, price: <50.00]`	0.7500	0.0000
1	`click[b09qcp4579]`	0.8000	0.0000
2	`click[purple]`	0.8000	0.0000
3	**`click[purple]`**	**0.4000**	**0.0000**
4	**`click[black]`**	**0.0000**	**0.0000**
5	**`click[black]`**	**0.0000**	**0.0000**
6	`click[buy now]`	–	0.0000
Outcome reward (failure)			**0.0000**

The direct reward distinguishes redundant and conflicting decisions, while return-to-go assigns zero to all actions because the final purchase fails.

WebShop B: Visible constraint violation

Task. Find silver open-toe women's flats in size 9 with a rubber sole below $60.
Key observation. The selected product already violates visible constraints before the product page is opened.

Step	Action	Direct	RTG
0	`search[open toe women's flats silver rubber sole size 9 <60]`	1.0000	6.9834
1	**`click[b09s6vn97v]`**	**0.0000**	**7.3509**
2	`click[back to search]`	1.0000	7.7378
3	`search[open toe women's flats silver rubber sole size 9 <60]`	0.7500	8.1451
4	`click[b09mfz726l]`	1.0000	8.5738
5	`click[silver]`	1.0000	9.0250
6	`click[9]`	1.0000	9.5000
7	`click[buy now]`	–	10.0000
Outcome reward (success)			**10.0000**

The incorrect action receives a positive return because later decisions recover the trajectory.

WebShop C: Useful decision followed by later failure

Task. Find a relaxed-fit men's polo in `military green`, size `x-large`, below $60.
Key observation. The selected product is a reasonable candidate, but later inspection reveals that the required color is unavailable.

Step	Action	Direct	RTG
0	`search[mens polos relaxed fit military green x-large price]`	1.0000	0.0000
1	`click[b01mg1ltms]`	0.0000	0.0000
2	`click[x-large]`	1.0000	0.0000
3	`click[military green]`	1.0000	0.0000
4	`click[back to search]`	1.0000	0.0000
5	`search[mens polos relaxed fit military green x-large price]`	0.7500	0.0000
6	**`click[b09qqp3356]`**	**1.0000**	**0.0000**
7	`click[x-large]`	1.0000	0.0000
8	`click[buy now]`	–	0.0000
Outcome reward (failure)			**0.0000**

The product selection is locally reasonable at decision time, but receives zero return because the final trajectory fails.

D.5.2 ALFWORLD

ALFWorld A: Inspection before available placement

Task. Put a clean egg in the sidetable.
Key observation. After cleaning the egg and reaching the target location, the required placement action is already available.

Step	Action	Direct	RTG
0	`go to fridge 1`	1.0000	6.9834
1	`open fridge 1`	1.0000	7.3509
2	`take egg 1 from fridge 1`	1.0000	7.7378
3	`go to sinkbasin 1`	1.0000	8.1451
4	`clean egg 1 with sinkbasin 1`	1.0000	8.5737
5	`go to sidetable 1`	1.0000	9.0250
6	**`examine egg 1`**	**0.5000**	**9.5000**
7	`move egg 1 to sidetable 1`	1.0000	10.0000
Outcome reward (success)			**10.0000**

The inspection is unnecessary at this point, but receives high return because the final action completes the task.

ALFWorld B: Exploration without task progress

Task. Heat a cup and put it in the fridge.
Key observation. The agent holds the required cup but performs several unrelated exploration actions before reaching the microwave.

Step	Action	Direct	RTG
0	`go to diningtable 1`	1.0000	2.2594
1	`take cup 1 from diningtable 1`	1.0000	2.3783
2–15	*Search and navigation actions before heating*	–	–
16	**`open cabinet 1`**	**0.5000**	**5.1334**
17	`go to cabinet 2`	0.0000	5.4036
18	**`open cabinet 2`**	**0.5000**	**5.6880**
19–24	*Additional exploration before heating*	–	–
25	`go to microwave 1`	1.0000	8.1451
26	`open microwave 1`	1.0000	8.5737
27	`heat cup 1 with microwave 1`	1.0000	9.0250
28	`go to fridge 1`	1.0000	9.5000
29	`move cup 1 to fridge 1`	1.0000	10.0000
Outcome reward (success)			**10.0000**

The exploration actions receive positive return because they occur before successful completion, while their local usefulness is limited.

For all examples, R_t is computed from the recorded terminal reward with $\gamma = 0.95$. The tables report raw direct process rewards and return-to-go values before conversion into state-conditioned relative advantages. The episode-level advantage is omitted.