# REWARDWEAVER: LONG-HORIZON INTERACTIVE LEARNING FOR LANGUAGE AGENTS VIA SELF-EVOLVING REWARD ADAPTATION

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Reinforcement learning with verifiable rewards (RLVR) has driven substantial progress in domains where task outcomes can be reliably evaluated, but long-horizon interaction remains challenging due to sparse terminal feedback and difficult credit assignment. Process rewards provide denser supervision, yet the capabilities most relevant for training can change as the policy evolves: a behavior that is easy to evaluate or frequently deficient need not be the bottleneck currently limiting task success. We introduce **RewardWeaver**, a self-evolving reward adaptation framework for language agents in long-horizon interaction. RewardWeaver maintains a validated capability space in which the semantics of admitted Rubrics remain fixed, and closes the loop between policy optimization, task evaluation, failure attribution, and reward adaptation. After each training stage, it performs outcome-grounded backward attribution on low-outcome trajectories, aggregates recurrent and policy-controlled capability bottlenecks, and dynamically selects the corresponding process rewards for the next stage. Recurrent failures not covered by the existing capability space trigger a separate, controlled expansion procedure. We evaluate REWARDWEAVER on SOTOPIA, AmazonHistoryPrice, and a newly constructed Sales Benchmark. Across social interaction, bilateral bargaining, and domain-specific sales, REWARDWEAVER establishes new state-of-the-art (SOTA) results. Ablations further demonstrate the importance of dynamic reward allocation, failure-grounded attribution, and stable semantics for admitted capabilities.

## 1 INTRODUCTION

Reinforcement learning with verifiable rewards (RLVR) has enabled substantial progress in mathematics and code, where task outcomes can be reliably checked by rules, tests, or execution (Shao et al., 2024; Le et al., 2022; Guo et al., 2025). Long-horizon interaction, however, poses a fundamentally different optimization problem. In settings such as negotiation, social coordination, and collaborative decision making (Zhou et al., 2024a; Wang et al., 2026; Liu et al., 2025), each action changes the partner's subsequent behavior and reshapes the interaction trajectory. Final outcomes emerge from sequences of interdependent decisions across many turns, making rewards sparse, delayed, and difficult to assign to specific behaviors (Arjona-Medina et al., 2019; Hung et al., 2019; Meulemans et al., 2023). A terminal score indicates whether an interaction succeeded, but certifies failure without localizing its cause and cannot by itself direct training.

Rubric-based rewards and evaluators provide denser supervision by assessing intermediate behaviors against explicit, fine-grained criteria (Gunjal et al., 2026; Wei et al., 2026; Hashemi et al., 2024; Lee et al., 2025), and rubric construction now scales from instance-specific generation to reusable repositories (He et al., 2025; Li et al., 2026). Recent approaches further revise or evolve these criteria as the policy changes, tracking its current response distribution (Guan et al., 2026). However, tracking the response distribution is not tracking the failure distribution: a behavior may be frequent or easy to discriminate without being a recurrent cause of task failure, while a less frequent capability deficiency may constitute the actual performance bottleneck. The key question is not only how evaluation criteria should adapt, but what evidence should drive that adaptation. We argue that this evidence should be recurrent failures in low-outcome trajectories, rather than response characteristics alone. Figure 1 contrasts these reward-adaptation paradigms.



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.

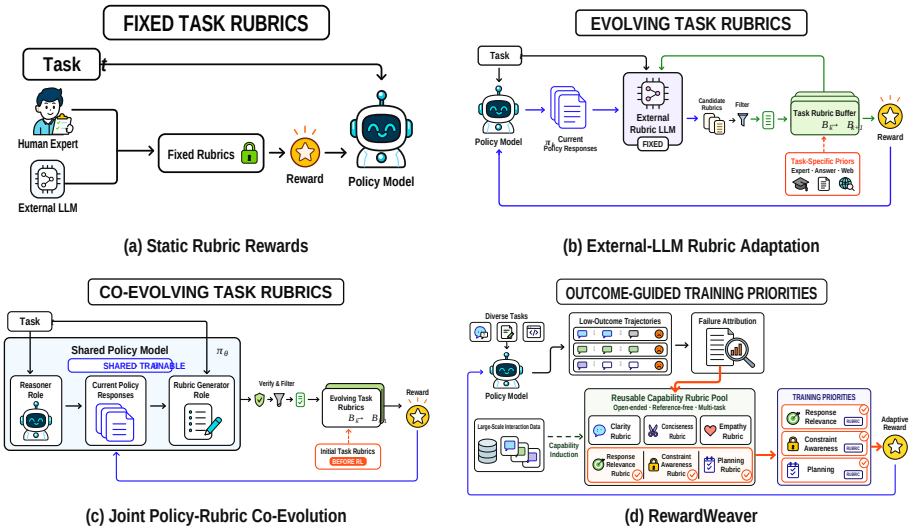


Figure 1: **Reward Adaptation under Policy Evolution.** **(a)** Task-specific Rubrics remain fixed throughout training. **(b)** External-LLM methods revise Rubrics as policy evolves. **(c)** Co-evolution methods jointly update Rubrics and the policy. **(d)** RewardWeaver grounds reward adaptation in recurrent task failures and prioritizes capability Rubrics according to current policy bottlenecks.

The central challenge is not merely how to design better process rewards, but how to decide what the current policy should optimize next. This introduces an outer-loop optimization problem complementary to standard RL: the inner-loop optimizer determines how to improve the policy under a given reward, while the outer loop must determine which capabilities deserve training credit. Diagnostic evidence must therefore satisfy three conditions: it must be outcome-grounded, drawn from trajectories that failed rather than sampled uniformly; recurrent, reappearing across trajectories rather than reflecting incidental variation; and policy-controlled, attributable to the agent's own behavior (Zhang et al., 2025a) rather than to the partner or the scenario. Isolated low scores satisfy none of these conditions. In this view, the task-level objective should remain stable, while process-level training credit shifts toward the capabilities that currently constrain performance.

We introduce **RewardWeaver**, a self-evolving reward adaptation framework for language agents in long-horizon interaction. REWARDWEAVER closes the loop between policy training, automatic evaluation, failure attribution, and reward adaptation. At each stage, it identifies recurrent capability bottlenecks from on-policy interaction failures and dynamically selects the corresponding Rubrics to construct the next process reward. Unlike prior self-evolving rubric methods that evolve *what is evaluated*, REWARDWEAVER evolves *what is optimized*: it preserves stable semantics for admitted capabilities while dynamically reallocating training credit across them. Recurrent failures that fall outside the existing Rubric space are incorporated only through controlled validation, allowing the reward to evolve without uncontrolled semantic drift.

We evaluate REWARDWEAVER on SOTOPIA (Zhou et al., 2024a) and related social intelligence benchmarks against outcome-only reinforcement learning, static capability prioritization, and alternative attribution and rubric-design strategies. REWARDWEAVER improves the primary evaluation metrics and achieves new state-of-the-art (SOTA) performance under matched training and evaluation protocols. Further ablation, sensitivity, and qualitative analyses show that failure-grounded attribution, adaptive credit reallocation, and stable semantics for admitted capabilities are key contributors to the gains. Our contributions are summarized as follows: (1) We formulate reward adaptation in long-horizon interaction as an outer-loop optimization problem that determines which process capabilities should be prioritized next. (2) We propose REWARDWEAVER, which dynamically reallocates training credit through failure-driven capability-bottleneck identification while preserving stable task objectives and evaluation semantics. (3) We conduct systematic experiments demonstrating the advantages of self-evolving reward adaptation over outcome-only and static process-reward baselines, and isolate the contributions of REWARDWEAVER's key components.

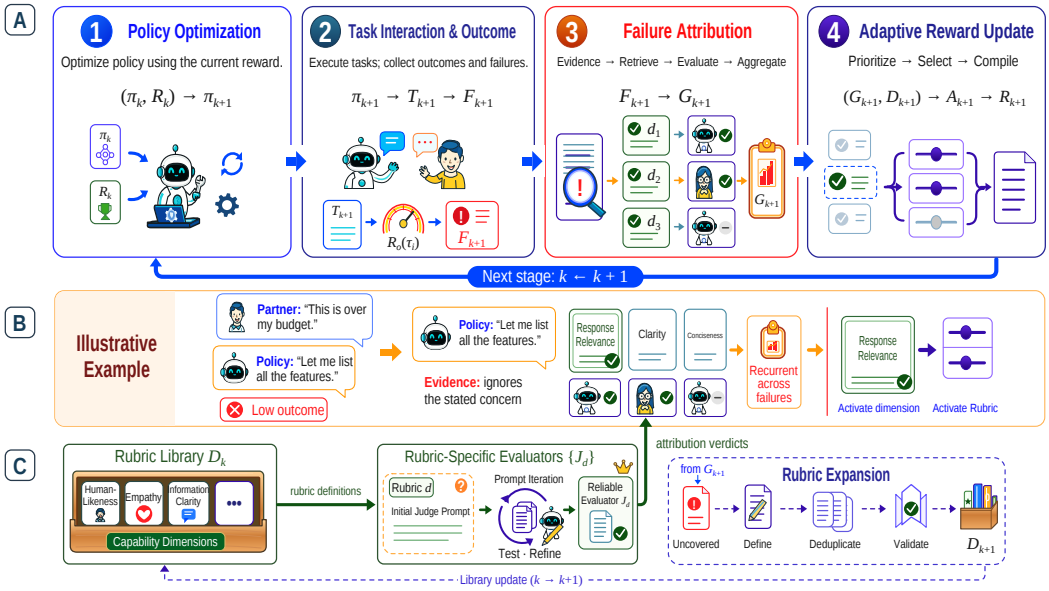


Figure 2: **The Overall Architecture of REWARDWEAVER. (a)** The policy is optimized under the current reward; its on-policy low-outcome trajectories reveal recurrent capability bottlenecks that determine the next-stage allocation of training credit. **(b)** Failure attribution grounds bottleneck identification in concrete behavioral evidence. **(c)** A validated capability space provides stable semantics for admitted capabilities, while recurrent uncovered failures trigger controlled expansion.

## 2 REWARDWEAVER METHODOLOGY

REWARDWEAVER separates *reward semantics* from *reward relevance*: a validated capability space determines *what can be rewarded*, while a stage-wise outer loop determines *what should be rewarded*. Training priorities can evolve with the policy without the semantics of admitted capabilities drifting, while the capability library itself expands only through a separate validated procedure.

### 2.1 TASK FORMULATION

At stage $k$, policy $\pi_k$ interacts with partner $\pi_{\text{partner}}$: given interaction history $h_t$, the two parties act alternately with $a_t \sim \pi_k(\cdot \mid h_t)$ and $b_t \sim \pi_{\text{partner}}(\cdot \mid h_t, a_t)$. A completed trajectory $\tau$ receives a fixed task-level outcome $R_o(\tau)$ and a stage-dependent process reward,

$$R_k(\tau) = R_o(\tau) + \frac{1}{|A_k|} \sum_{d \in A_k} \bar{J}_d(\tau), \tag{1}$$

where $\mathcal{D}_k$ is the capability space available at stage $k$, $A_k \subseteq \mathcal{D}_k$ the active subset, and $\bar{J}_d(\tau)$ the trajectory-level score of capability evaluator $J_d$; all active Rubrics enter with equal weight, so adaptation determines *which* capabilities receive credit rather than how much each receives. The terminal objective remains fixed; only process-level credit changes: standard RL determines *how* to optimize under $R_k$, i.e., $(\pi_k, R_k) \to \pi_{k+1}$, whereas REWARDWEAVER determines *which capabilities should receive training credit next*. Throughout, *capability* denotes the behavioral concept, *Rubric* its frozen operational definition, and *evaluator* the scoring function $J_d$; the *active set* is the subset of capabilities currently receiving process-level training credit, and a *reward specification* couples the active set with how its evaluators are composed.

### 2.2 VALIDATED CAPABILITY SPACE

Reward adaptation requires stable semantics for admitted capabilities. REWARDWEAVER operates over a validated space $\mathcal{D}$ of observable, policy-controllable behaviors, where each $d \in \mathcal{D}$ couples a frozen Rubric definition with an evaluator $J_d(x) = M_{\text{eval}}(p_d, x)$, and $p_d$ operationalizes

**Algorithm 1 Validated Capability-Space Construction**

**Require:** Raw interaction corpus $\mathcal{T}_{\text{raw}}$, induction models $\mathcal{M}$, validity criteria $\mathcal{V}$, evaluator configuration $\kappa$, repeat count $R$, construction budget $B$
**Ensure:** Validated capability space $\mathcal{D} = \{(d, J_d)\}$

**Stage I: Capability Discovery and Consolidation**

$\mathcal{T} \leftarrow \text{NORMALIZEANDFILTER}(\mathcal{T}_{\text{raw}})$
$\mathcal{F} \leftarrow \text{OPENINDUCTIONWITHEVIDENCE}(\mathcal{T}, \mathcal{M})$
**for** $r = 1, \ldots, R$ **do**
  $\mathcal{S}_r \leftarrow \text{CONSOLIDATE}(\text{EMBEDDINGNEIGHBORS}(\mathcal{F}))$
  $\mathcal{S}_r \leftarrow \text{FILTER}(\mathcal{S}_r; \text{generality}, \text{operationality}, \text{distinctiveness})$
**end for**
$\mathcal{S} \leftarrow \text{ALIGNANDSELECT}(\{\mathcal{S}_r\}_{r=1}^{R}; \text{recurrence}, \text{coverage}, \text{support})$
$\mathcal{S} \leftarrow \text{DEFINECANDIDATES}(\mathcal{S}, \mathcal{F})$

**Stage II: Operationalization and Admission**

$\mathcal{D} \leftarrow \varnothing$
**while** $\mathcal{S} \neq \varnothing$ and budget $B$ remains **do**
  remove candidate $d$ from $\mathcal{S}$
  $\mathcal{X}_d \leftarrow \text{RETRIEVERELEVANTCASES}(\mathcal{T}, d)$
  $\mathcal{L}_d \leftarrow \text{ANNOTATEANDVERIFY}(\mathcal{X}_d, d)$
  $(\mathcal{L}_d^{\text{dev}}, \mathcal{L}_d^{\text{acc}}) \leftarrow \text{SOURCEDISJOINTSPLIT}(\mathcal{L}_d)$
  $(\mathcal{L}_d^{\text{dev}}, \mathcal{L}_d^{\text{acc}}) \leftarrow \text{ENSURECOVERAGE}(\mathcal{L}_d^{\text{dev}}, \mathcal{L}_d^{\text{acc}}, d)$
  $p_d^\star \leftarrow \text{DEVELOPEVALUATOR}(d, \mathcal{L}_d^{\text{dev}}, \kappa)$
  $J_d \leftarrow \text{FREEZEEVALUATOR}(p_d^\star, d, \kappa)$
  $v_d \leftarrow \text{VALIDATECANDIDATE}(d, J_d, \mathcal{L}_d^{\text{acc}}, \mathcal{D}; \mathcal{V})$
  **if** $v_d = \text{ACCEPT}$ **then**
    $\mathcal{D} \leftarrow \mathcal{D} \cup \{(\text{FREEZEDEFINITION}(d), J_d)\}$
  **else if** $v_d = \text{REVISE}$ **then**
    $d' \leftarrow \text{REVISECANDIDATE}(d, v_d)$
    $\mathcal{S} \leftarrow \mathcal{S} \cup \{d'\}$
  **else**
    discard $d$
  **end if**
**end while**
**return** $\mathcal{D}$

the definition over interaction context $x$. Construction has two stages: candidates are first induced from interaction evidence, consolidated across runs, and filtered for generality, operationality, and semantic distinctiveness; each is then operationalized by a dedicated evaluator and subjected to source-disjoint acceptance testing (Algorithm 1). Candidates may be revised before admission; only validated pairs are admitted into $\mathcal{D}$, so discovery remains flexible while admitted semantics remain fixed. Admission testing certifies each evaluator on its own source-disjoint acceptance set, so every admitted capability satisfies the same validity criteria. The SOTOPIA capability space, representative entries, and their operationalization into executable criteria are provided in Appendix A.

### 2.3 Outcome-Grounded Failure Attribution

Terminal outcomes identify low-outcome trajectories that warrant diagnosis, but do not by themselves establish an agent-side failure or determine what should receive training credit. After stage $k$, $\pi_{k+1}$ generates on-policy trajectories $\mathcal{T}_{k+1}$, from which the low-outcome cohort $\mathcal{F}_{k+1} = \{\tau_i \in \mathcal{T}_{k+1} \mid q_o(R_o(\tau_i), x_i) = 1\}$ is selected, where $x_i$ denotes the task instance of trajectory $\tau_i$. For each $\tau_i \in \mathcal{F}_{k+1}$, REWARDWEAVER localizes failure-relevant evidence, retrieves candidate capabilities $C_i \subseteq \mathcal{D}_k$, and produces attribution records $a_{i,d} = (s_{i,d}, c_{i,d}, e_{i,d})$ for $d \in C_i$, where $s_{i,d}$ is the deficit judgment, $c_{i,d}$ its confidence, and $e_{i,d}$ the supporting evidence; unsupported assignments are rejected. Trajectory-level records are aggregated into $G_{k+1} = \text{Aggregate}(\{a_{i,d}\})$, which retains recurrent, supported, and policy-controlled failure modes rather than isolated low scores.

### 2.4 Adaptive Credit Allocation

A capability can be valid yet irrelevant to the current policy; REWARDWEAVER treats training relevance as stage-dependent. Given $G_{k+1}$, it selects a budgeted active set $A_{k+1} =$

**Algorithm 2 Failure Attribution and Adaptive Credit Allocation**

**Require:** On-policy trajectories $\mathcal{T}_{k+1}$, capability space $\mathcal{D}_k$, current reward specification $P_k$, adaptation history $\mathcal{S}_k$, task-performance summary $y_{k+1}$, calibration set $\mathcal{H}_{k+1}$, parameters $\Theta$ with active budget $K$ and calibration limit $M$

**Ensure:** Next-stage reward specification $P_{k+1}$, or calibration failure $\perp$

$\mathcal{F}_{k+1} \leftarrow \textsc{LowOutcomes}(\mathcal{T}_{k+1})$; $\mathcal{Z}_{k+1} \leftarrow \varnothing$
**for** $\tau \in \mathcal{F}_{k+1}$ **do**
  $e_\tau \leftarrow \textsc{LocalizeEvidence}(\tau)$
  $C_\tau \leftarrow \textsc{Retrieve}(e_\tau, \mathcal{D}_k)$
  $\mathcal{Z}_\tau \leftarrow \textsc{Attribute}(e_\tau, C_\tau)$ ▷ Abstention allowed
  $\mathcal{Z}_{k+1} \leftarrow \mathcal{Z}_{k+1} \cup \mathcal{Z}_\tau$
**end for**
$(G_{k+1}, \{r_d, n_d, c_d, f_d\}) \leftarrow \textsc{Aggregate}(\mathcal{Z}_{k+1})$
$p_d \leftarrow \alpha r_d + \beta c_d + \gamma(1 - f_d), \quad d \in \mathcal{D}_k$
$(\mathcal{B}^{\text{high}}, \mathcal{B}^{\text{weak}}) \leftarrow \textsc{Eligible}(G_{k+1}, \mathcal{S}_k; \Theta)$
$A_{k+1} \leftarrow \textsc{TopK}(\mathcal{B}^{\text{high}}, p, K)$
**if** $|A_{k+1}| < K$ **and** $\textsc{AllowWeak}(\mathcal{B}^{\text{high}}, \mathcal{S}_k; \Theta)$ **then**
  $A_{k+1} \leftarrow A_{k+1} \cup \textsc{TopK}(\mathcal{B}^{\text{weak}} \setminus A_{k+1}, p, K - |A_{k+1}|)$
**end if**
$(A_{k+1}, \mathcal{S}_{k+1}) \leftarrow \textsc{Schedule}(A_{k+1}, G_{k+1}, \mathcal{S}_k, y_{k+1}; \Theta)$
$\lambda_{k+1,d} \leftarrow 1/|A_{k+1}|, \quad d \in A_{k+1}$ ▷ Equal weight over active Rubrics
$Q_{k+1} \leftarrow \textsc{CompileRubrics}(A_{k+1}, P_k)$ ▷ Assemble a reward specification
**for** $m = 1, \ldots, M$ **do**
  $(s, \delta) \leftarrow \textsc{Calibrate}(Q_{k+1}, \mathcal{H}_{k+1})$
  **if** $\textsc{Accept}(s)$ **then** ▷ Rubric-set publication gate
    $P_{k+1} \leftarrow \textsc{Compose}(P_k, A_{k+1}, \lambda_{k+1}, Q_{k+1}, s, \delta, \mathcal{S}_{k+1})$
    $\textsc{Publish}(P_{k+1})$
    **return** $P_{k+1}$
  **else if** $m < M$ **then**
    $Q_{k+1} \leftarrow \textsc{Revise}(Q_{k+1}, s, \delta)$ ▷ Revise composition; keep $(d, J_d)$ frozen
  **end if**
**end for**
**return** $\perp$ ▷ Calibration failure; no new reward is published

$\text{Select}(G_{k+1}, \mathcal{D}_k) \subseteq \mathcal{D}_k$. All active Rubrics contribute equally to the process reward, yielding $R_{k+1}(\tau) = R_o(\tau) + \frac{1}{|A_{k+1}|} \sum_{d \in A_{k+1}} \bar{J}_d(\tau)$. As a bottleneck recedes, its Rubric may leave the active set and capacity is released for newly dominant failures, so the process reward is selective and stage-dependent rather than cumulative. Letting $P_k$ denote the deployed process-reward specification at stage $k$, $\mathcal{S}_k$ its adaptation history, and $y_{k+1}$ the stage-level task-performance summary, Algorithm 2 specifies the complete attribution-to-reward update. Importantly, the update changes which Rubrics are active and how they are composed into the next-stage reward, but never modifies the frozen semantics or evaluators of the capabilities admitted to $\mathcal{D}_k$.

### 2.5 Controlled Capability Expansion

Adaptive credit allocation operates only over the capability space already validated at the beginning of the update, so the transition from stage $k$ to $k+1$ always satisfies $A_{k+1} \subseteq \mathcal{D}_k$. When recurrent failures cannot be represented by $\mathcal{D}_k$, RewardWeaver invokes the same admission procedure as Algorithm 1: validated additions form the expanded space $\mathcal{D}_{k+1}$, but newly admitted capabilities do not alter the reward update that triggered their discovery—they become eligible only in the subsequent adaptation step, i.e., $A_{k+2} \subseteq \mathcal{D}_{k+1}$. The fast loop therefore reallocates training credit within fixed semantics,

$$(\pi_k, R_k) \to \pi_{k+1} \to \mathcal{T}_{k+1} \to \mathcal{F}_{k+1} \to G_{k+1} \to A_{k+1} \subseteq \mathcal{D}_k \to R_{k+1}, \tag{2}$$

while the slower semantic loop updates $(\mathcal{F}_{k+1}, \mathcal{D}_k) \to \mathcal{D}_{k+1}$, with any new dimensions becoming available only to later reward updates. This separation defines the methodological boundary of RewardWeaver: *training priorities evolve rapidly; reward semantics evolve conservatively*.

Table 1: **Main results on SOTOPIA.** The best result in each setting is shown in **bold**. In the Partner setting, the counterpart is GPT-4o for literature results and Qwen2.5-72B for AMPO and REWARDWEAVER in our experiments.

| Method | Self-Play | | Partner | |
|---|---|---|---|---|
| | SOTOPIA Goal / Overall ↑ | SOTOPIA-Hard Goal / Overall ↑ | SOTOPIA Goal / Overall ↑ | SOTOPIA-Hard Goal / Overall ↑ |
| *Proprietary LLMs* | | | | |
| GPT-4o | 8.19 / 3.76 | 6.97 / 3.46 | 8.19 / 3.76 | 6.97 / 3.46 |
| Claude-3.5-Sonnet | 8.29 / 3.71 | 6.33 / 3.09 | 8.42 / 3.77 | 6.64 / 3.30 |
| DeepSeek-V3 | 8.15 / 3.62 | 6.34 / 3.09 | 8.14 / 3.72 | 6.69 / 3.31 |
| *Large Reasoning Models* | | | | |
| OpenAI-o1 | 7.93 / 3.58 | 5.69 / 2.71 | 8.09 / 3.69 | 6.65 / 3.20 |
| OpenAI-o3-mini | 7.38 / 3.30 | 5.14 / 2.36 | 7.96 / 3.61 | 6.33 / 2.98 |
| Gemini-2.5-Pro | 7.85 / 3.43 | 5.67 / 2.55 | 8.12 / 3.59 | 6.70 / 3.09 |
| DeepSeek-R1 | 7.97 / 3.40 | 5.86 / 2.73 | 7.92 / 3.49 | 6.20 / 2.95 |
| QwQ-32B | 7.70 / 3.30 | 5.35 / 2.41 | 7.80 / 3.47 | 6.19 / 2.91 |
| *Post-training on Qwen2.5-7B-Instruct* | | | | |
| Qwen2.5-7B-Instruct | 7.91 / 3.55 | 6.21 / 3.01 | 6.71 / 3.13 | 5.90 / 2.90 |
| w/ PPDPP (Deng et al., 2024) | 7.97 / 3.65 | 6.63 / 3.31 | 8.07 / 3.71 | 6.76 / 3.35 |
| w/ EPO (Liu et al., 2025) | 8.09 / 3.51 | 6.82 / 3.12 | 8.41 / 3.86 | 6.81 / 3.51 |
| w/ DAT (Li et al., 2024) | 7.97 / 3.59 | 6.39 / 3.10 | 8.11 / 3.70 | 6.78 / 3.36 |
| w/ DSI (Zhang et al., 2025b) | 8.35 / 3.75 | 7.31 / 3.51 | 8.15 / 3.70 | 6.87 / 3.42 |
| AMPO (Wang et al., 2026) | 8.45 / 4.20 | 7.67 / 3.94 | 8.30 / 4.09 | 7.30 / 3.80 |
| **REWARDWEAVER (ours)** | **8.72 / 4.47** | **7.90 / 4.38** | **8.49 / 4.47** | **7.66 / 4.22** |

# 3 EXPERIMENTS

## 3.1 EXPERIMENTAL SETTINGS

**Benchmarks.** We evaluate REWARDWEAVER on three long-horizon benchmarks that together span general social interaction, bilateral bargaining, and domain-specific sales. SOTOPIA (Zhou et al., 2024a) contains 90 social scenarios; we report on the full benchmark and SOTOPIA-Hard. Amazon-HistoryPrice (Xia et al., 2024) contains 930 products across 18 categories and evaluates bargaining under asymmetric information, with Mutual Interest (MI) instances admitting feasible agreements and Conflicting Interest (CI) instances admitting none. Following SOTOPIA-$\pi$ (Wang et al., 2024b), we construct a health-service Sales Benchmark of 400 tasks across 20 scenario families; evaluation uses held-out instances within training families (see Appendix B).

**Evaluation.** SOTOPIA and the Sales Benchmark use the SOTOPIA-EVAL protocol (Zhou et al., 2024a), reporting Goal Completion and Overall under Self-Play and fixed-Partner settings; SOTOPIA-Hard is the challenging subset, and Sales-Hard the 50 constraint-conflict tasks. For AmazonHistoryPrice, we report MI Deal Rate, CI Correct No-Deal Rate, Task Success Rate, Normalized Seller Utility, and Expected MI Seller Return. Task Success counts a legal MI agreement or correct CI no-deal; Expected MI Seller Return combines agreement probability with the resulting seller utility. For SOTOPIA, REWARDWEAVER uses a validated 40-dimensional capability space (Appendix A), activates at most $B = 6$ Rubrics per stage, and updates allocation every 30 steps.

## 3.2 EXPERIMENTAL RESULTS AND ANALYSIS

**RQ1: Does RewardWeaver improve long-horizon interactive learning?** Across the three benchmarks, REWARDWEAVER improves most primary metrics, with particularly strong gains in the harder SOTOPIA settings. On SOTOPIA (Table 1), it outperforms AMPO, the strongest prior method, on every metric under the same partner protocol, most pronounced on SOTOPIA-Hard (Goal/Overall: 7.67/3.94 to 7.90/4.38 Self-Play, 7.30/3.80 to 7.66/4.22 Partner), suggesting adaptive reward allocation remains effective under harder scenarios and external partners. On Amazon-HistoryPrice (Figure 3), REWARDWEAVER jointly improves MI Deal Rate and seller-side utility,

reaching the upper-right performance frontier; complete numerical results are reported in Appendix Table C1. On the Sales Benchmark (Table C2), it achieves the best result on six of eight metrics.

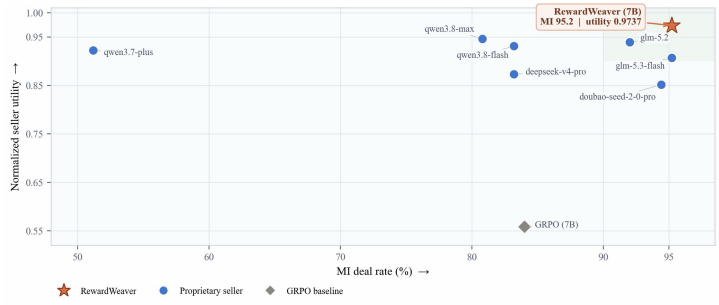


Figure 3: **Main results on AmazonHistoryPrice.** Under the DeepSeek-V4-Pro buyer, REWARDWEAVER reaches the upper-right frontier, achieving a 95.2% MI Deal Rate and the highest Normalized Seller Utility (0.9737) among all evaluated methods.

**RQ2: Does dynamic reward adaptation matter, and does it need to target the correct bottlenecks?** Table 2 compares REWARDWEAVER with outcome-only GRPO, Static Top-$K$, and Random Active Set. Outcome-only GRPO is weaker overall, supporting the usefulness of process-level supervision in this setting, while Static Top-$K$ remains below REWARDWEAVER, suggesting that a fixed capability set becomes insufficient as training proceeds. Random Active Set preserves stage-wise reward changes but breaks the mapping from diagnosed bottlenecks to rewarded capabilities, providing a controlled test of bottleneck alignment; it underperforms REWARDWEAVER across all eight metrics. This comparison supports the importance of aligning reward allocation with the diagnosed bottlenecks, beyond the benefit of reward changes alone.

Table 2: **Ablation study on SOTOPIA.** The counterpart is Qwen2.5-72B. The best result in each column is **bold**. We isolate the effects of dynamic reward adaptation, failure-grounded bottleneck identification, and stable semantics for admitted capabilities under matched training budgets.

| Method | Self-Play | | Partner | |
|---|---|---|---|---|
| | SOTOPIA Goal / Overall ↑ | SOTOPIA-Hard Goal / Overall ↑ | SOTOPIA Goal / Overall ↑ | SOTOPIA-Hard Goal / Overall ↑ |
| **REWARDWEAVER (ours)** | **8.72** / 4.47 | 7.90 / **4.38** | **8.49 / 4.47** | 7.66 / **4.22** |
| *Dynamic Reward Adaptation* | | | | |
| Outcome-only GRPO | 8.43 / 4.31 | 7.53 / 4.05 | 8.35 / 4.26 | 7.71 / 4.12 |
| Static Top-$K$ Dimensions | 8.44 / 4.31 | 7.77 / 4.14 | 8.36 / 4.23 | 7.49 / 3.83 |
| Random Active Set | 8.47 / 4.41 | 7.56 / 4.18 | 8.33 / 4.27 | 7.31 / 3.97 |
| *Failure-Grounded Bottleneck Identification* | | | | |
| Dynamic Top-$K$ by Failure Frequency | 8.21 / 4.23 | 7.36 / 4.03 | 8.11 / 3.40 | 7.13 / 3.25 |
| Random-Trajectory Attribution | 8.38 / 4.37 | 7.67 / 4.25 | 8.24 / 4.31 | 7.60 / 4.17 |
| Success–Failure Trajectory Attribution | 8.55 / **4.52** | **7.93** / 4.32 | 8.34 / 4.33 | 7.60 / 4.14 |
| *Stable Semantics for Admitted Capabilities* | | | | |
| Open-Vocabulary Dimensions | 8.50 / 4.37 | 7.69 / 4.19 | 8.35 / 4.18 | 7.44 / 3.95 |
| Instance-Specific Rubrics | 8.61 / 4.43 | 7.90 / 4.29 | 8.47 / 4.33 | **7.83** / 4.21 |

**RQ3: Is failure-grounded attribution necessary for identifying policy bottlenecks?** Table 2 separates failure attribution from simpler deficiency-based selection. Dynamic Top-$K$ by Failure Frequency reselects capabilities each stage by deficiency prevalence on low-outcome trajectories, but performs no backward attribution or evidence localization. It underperforms REWARDWEAVER across all eight metrics, suggesting deficiency frequency alone is insufficient to reliably identify failure-relevant capabilities in these settings. Random-Trajectory Attribution degrades performance,

suggesting attribution needs to be grounded in outcome-relevant failures rather than generic behavior. Success–Failure Trajectory Attribution remains competitive but does not consistently improve over REWARDWEAVER. These results suggest training relevance is better identified by evidence-grounded failure attribution than by deficiency prevalence or outcome-agnostic behavioral analysis. Appendix D provides trajectory-level cases of contrastive validation and selective attribution.

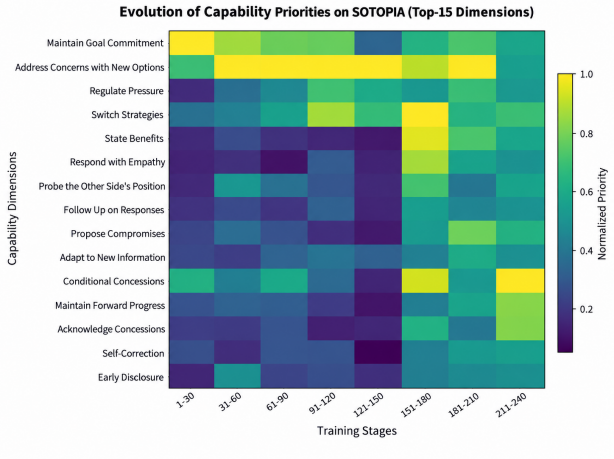


Figure 4: **Evolution of capability priorities on SOTOPIA.** Normalized priorities of the 15 most prominent Rubrics across training stages. Reward emphasis shifts as different capability bottlenecks become training-relevant, illustrating stage-dependent reallocation of process-level credit in REWARDWEAVER.

**RQ4: Does semantic stability of admitted capabilities matter?** As shown in Table 2, relaxing semantic stability also reduces performance. Open-Vocabulary Dimensions is consistently weaker than REWARDWEAVER across all four settings, suggesting an advantage of maintaining stable, validated capability definitions. Instance-Specific Rubrics is competitive, but remains inconsistent across settings despite occasionally matching or exceeding individual metrics.

### 3.3 REWARD ADAPTATION ANALYSIS

**Reward Evolution.** Figure 4 shows reward emphasis shifting across training stages. *Maintain Goal Commitment* dominates early but later recedes, while *Switch Strategies*, *State Benefits*, and *Respond with Empathy* become more prominent; other capabilities, such as *Conditional Concessions*, re-emerge after decline. These patterns show REWARDWEAVER reallocates process-level credit as training-relevant bottlenecks change, rather than accumulating reward Rubrics.

**Bottleneck Attribution and Subsequent Failure Reduction.** Complementing the controlled RQ2 ablations, we examine whether the attribution signal is prospectively associated with bottleneck reduction. We define bottleneck importance $I_{k,d} = F^{\text{fail}}_{k,d} - F^{\text{succ}}_{k,d}$ and next-stage failure reduction $\Delta F_{k,d} = F^{\text{fail}}_{k,d} - F^{\text{fail}}_{k+1,d}$. Bottleneck importance positively correlates with subsequent reduction (Spearman $\rho = 0.28$), remains positive after controlling for failure rate (partial $\rho = 0.22$), and strengthens within stage ($\rho = 0.31$), while current failure rate alone yields $\rho = 0.21$. The attribution ranking carries prospective information about which bottlenecks recede next. This analysis is not a causal identification of capability-level improvement; causal evidence comes from the controlled ablations in RQ2 and RQ3. Full results are reported in Appendix E.

**Sensitivity Analysis.** We vary both the active Rubric budget $B$ and the adaptation interval. Performance is non-monotonic in both: $B = 6$ achieves the strongest overall results among the tested budgets, while the 30-step interval provides the most consistently strong performance across Self-Play and Partner evaluation. Effective reward adaptation therefore depends on both *selective credit allocation* and an *appropriate adaptation timescale*. Full results are reported in Appendix F.

**Capability Retention.** We examine whether capabilities regress after leaving the active reward set, tracking semantically aligned capabilities with fixed evaluation criteria across stages. Across the six capabilities deactivated after the initial stage, all 36 post-deactivation observations remain within 0.02 of or above their final active scores, with a mean change of $+0.10$. *Adapt to New Information*, deactivated later, remains stable. On the fixed diagnostic set, we observe no systematic regression in the tracked capabilities after deactivation, providing evidence that reward reallocation need not immediately erase previously emphasized behaviors. Full results are reported in Appendix G.

## 4 Related Work

**Outcome and Preference-Based Rewards.** RLVR grounds rewards in verifiable task outcomes (Shao et al., 2024; Le et al., 2022; Guo et al., 2025), and process verifiers extend it to intermediate solution steps (Wang et al., 2024a; Lightman et al., 2024; Lu et al., 2024; Zeng et al., 2025; Setlur et al., 2025; Zheng et al., 2025). RLHF learns reward models from human preference data (Christiano et al., 2017; Stiennon et al., 2020; Ouyang et al., 2022); subsequent work diversifies feedback types (Wu et al., 2023; Yuan et al., 2024; Wang et al., 2024c), develops ranking objectives that bypass explicit reward modeling (Yuan et al., 2023; Hwang et al., 2023; Rafailov et al., 2023), and automates evaluation with AI feedback and pretrained LLM judges (Lee et al., 2023; Zheng et al., 2023; Liu et al., 2023; Kim et al., 2024a;b; Ke et al., 2024). In many of these approaches, however, the reward specification is fixed or largely determined before policy training; which capabilities the current policy should optimize next remains open.

**Rubric-Based Process Rewards.** Rubrics structure process quality: instance-specific rubrics are auto-generated (Gunjal et al., 2026; Wei et al., 2026; He et al., 2025), multidimensional evaluators induced (Hashemi et al., 2024), checklist scoring improves reliability (Lee et al., 2025), construction scales into reusable repositories (Li et al., 2026), and EvoRubric adapts criteria to the response distribution (Guan et al., 2026). These methods sharpen response-quality representation, yet discriminability does not determine training priority: a criterion may separate good from bad responses yet bear weakly on success, whereas a rarely violated capability may bind. Which criteria recurrent low-outcome failures implicate is underexplored.

**Credit Assignment in Multi-Turn Interaction.** Assigning feedback in long-horizon interaction is long-standing: methods decompose delayed returns (Arjona-Medina et al., 2019), transport value across time (Hung et al., 2019), or estimate counterfactuals (Meulemans et al., 2023). For language agents, feedback spans episode-level scoring (Zhou et al., 2024a), utterance-level rewards from episode feedback or Shapley decomposition (Yu et al., 2025; Feng et al., 2026), segment-level preferences (Kong et al., 2025), and process rewards over strategic reasoning (Liu et al., 2025); hierarchical and retrospective agents optimize behavior (Zhou et al., 2024b; Yao et al., 2024), and automated attribution localizes failures (Zhang et al., 2025a). Orthogonally, AMPO (Wang et al., 2026) adapts reasoning depth to interaction modes, WebRL (Qi et al., 2025) evolves curricula, and PPDPP (Deng et al., 2024), DAT (Li et al., 2024), and DSI (Zhang et al., 2025b) steer dialogue through planners or strategy-injected corpora. These methods determine *where* an objective applies within an interaction; reward adaptation asks *what* the process reward should contain; RewardWeaver selects capability-level objectives from outcome-grounded failures.

## 5 Discussion

The results of RewardWeaver suggest that reward relevance is stage-dependent and should be grounded in recurrent failures linked to task outcomes. This casts reward adaptation as an outer-loop problem: standard RL determines *how* to optimize under a given reward, while the outer loop determines *what should receive training credit next*. Reward adaptation should therefore track *what currently limits performance*, not merely *what can be evaluated well*.

RewardWeaver nevertheless depends on LLM-based evaluators and informative terminal outcomes, so errors in either can propagate into reward adaptation. Its active budget and adaptation interval may also require re-tuning across domains.

## AI USE STATEMENT

In this work, we used generative AI tools to support discussions of research methodology and experimental design, qualitative analysis and interpretation of results, and the drafting, restructuring, and language polishing of the manuscript. Generative AI was also used to improve the presentation and organization of selected algorithm descriptions, tables, figure captions, and appendix materials.

All AI-assisted content was carefully reviewed by the authors. The research methodology, experimental settings, data, and empirical results were verified against the actual implementation and experimental records, and the authors independently determined whether to adopt any AI-generated suggestions. The authors take full responsibility for the final content of this work, including any text, analysis, or other research artifacts generated or modified with the assistance of generative AI.

## REFERENCES

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 (NeurIPS)*, 2019.

Paul F. Christiano, Jan Leike, Tom B. Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

Yang Deng, Wenxuan Zhang, Wai Lam, See-Kiong Ng, and Tat-Seng Chua. Plug-and-play policy planner for large language model powered dialogue agents. In *International Conference on Learning Representations (ICLR)*, 2024.

Xiachong Feng, Yi Jiang, Xiaocheng Feng, Deyi Yin, Libo Qin, Yangfan Ye, Lei Huang, Weitao Ma, Yuxuan Gu, Chonghan Qin, Bing Qin, and Lingpeng Kong. SAVOIR: Learning social savoir-faire via shapley-based reward attribution. *arXiv preprint arXiv:2604.18982*, 2026.

Xin Guan, Xiaomeng Hu, Shen Huang, Zhenyi Wang, Bo Zhang, Zijian Li, Pengjun Xie, Bo Liu, and Jiuxin Cao. EvoRubric: Self-evolving rubric-driven RL for open-ended generation. *arXiv preprint arXiv:2605.29847*, 2026.

Anisha Gunjal, Anthony Wang, Elaine Lau, Vaskar Nath, Yunzhong He, Bing Liu, and Sean M. Hendryx. Rubrics as rewards: Reinforcement learning beyond verifiable domains. In *The Fourteenth International Conference on Learning Representations*, 2026.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Runxin Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. DeepSeek-R1 incentivizes reasoning in LLMs through reinforcement learning. *Nature*, 645:633–638, 2025.

Helia Hashemi, Jason Eisner, Corby Rosset, Benjamin Van Durme, and Chris Kedzie. LLM-Rubric: A multidimensional, calibrated approach to automated evaluation of natural language texts. In *Proceedings of ACL*, 2024.

Yun He, Wenzhe Li, Hejia Zhang, Songlin Li, Karishma Mandyam, Sopan Khosla, Yuanhao Xiong, Nanshu Wang, Xiaoliang Peng, Beibin Li, Shengjie Bi, Shishir G. Patil, Qi Qi, Shengyu Feng, Julian Katz-Samuels, Richard Yuanzhe Pang, Sujan Gonugondla, Hunter Lang, Yue Yu, Yundi Qian, Maryam Fazel-Zarandi, Licheng Yu, Amine Benhalloum, Hany Awadalla, and Manaal Faruqui. AdvancedIF: Rubric-based benchmarking and reinforcement learning for advancing LLM instruction following. *arXiv preprint arXiv:2511.10507*, 2025.

Chia-Chun Hung, Timothy Lillicrap, Josh Abramson, Yan Wu, Mehdi Mirza, Federico Carnevale, Arun Ahuja, and Greg Wayne. Optimizing agent behavior over long time scales by transporting value. *Nature Communications*, 2019.

Minyoung Hwang, Gunmin Lee, Hogun Kee, Chanwoo Kim, Kyungjae Lee, and Songhwai Oh. Sequential preference ranking for efficient reinforcement learning from human feedback. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Pei Ke, Bosi Wen, Zhuoer Feng, Xiao Liu, Xuanyu Lei, Jiale Cheng, Shengyuan Wang, Aohan Zeng, Yuxiao Dong, Hongning Wang, Jie Tang, and Minlie Huang. CritiqueLLM: Towards an informative critique generation model for evaluation of large language model generation. In *Proceedings of ACL*, 2024.

Seungone Kim, Jamin Shin, Yejin Cho, Joel Jang, Shayne Longpre, Hwaran Lee, Sangdoo Yun, Seongjin Shin, Sungdong Kim, James Thorne, and Minjoon Seo. Prometheus: Inducing fine-grained evaluation capability in language models. In *International Conference on Learning Representations (ICLR)*, 2024a.

Seungone Kim, Juyoung Suk, Shayne Longpre, Bill Yuchen Lin, Jamin Shin, Sean Welleck, Graham Neubig, Moontae Lee, Kyungjae Lee, and Minjoon Seo. Prometheus 2: An open source language model specialized in evaluating other language models. In *Proceedings of EMNLP*, 2024b.

Aobo Kong, Wentao Ma, Shiwan Zhao, Yongbin Li, Yuchuan Wu, Ke Wang, Xiaoqian Liu, Qicheng Li, Yong Qin, and Fei Huang. SDPO: Segment-level direct preference optimization for social agents. In *Proceedings of ACL*, pp. 12409–12423, 2025.

Hung Le, Yuheng Wang, Shafiq Gotmare, Silvio Savarese, and Steven Hoi. CodeRL: Mastering code generation through pretrained models and deep reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Harrison Lee, Samrat Phatale, Hassan Mansoor, Kellie Lu, Thomas Mesnard, Colton Bishop, Ethan Ferret, and Peter Riedel. RLAIF: Scaling reinforcement learning from human feedback with AI feedback. *arXiv preprint arXiv:2309.00267*, 2023.

Yukyung Lee, Joonghoon Kim, Jaehee Kim, Hyowon Cho, Pilsung Kang, and Najoung Kim. Check-Eval: A reliable LLM-as-a-judge framework for evaluating text generation using checklists. In *Proceedings of EMNLP*, 2025.

Kenneth Li, Yiming Wang, Fernanda Viégas, and Martin Wattenberg. Dialogue action tokens: Steering language models in goal-directed dialogue with a multi-turn planner. *arXiv preprint arXiv:2406.11978*, 2024.

Sunzhu Li, Jiale Zhao, Miteto Wei, Huimin Ren, Yang Zhou, Jingwen Yang, Shunyu Liu, Kaike Zhang, and Wei Chen. RubricHub: A comprehensive and highly discriminative rubric dataset via automated coarse-to-fine generation. *arXiv preprint arXiv:2601.08430*, 2026.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri 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 (ICLR)*, 2024.

Xiaoqian Liu, Ke Wang, Yongbin Li, Yuchuan Wu, Wentao Ma, Aobo Kong, Fei Huang, Jianbin Jiao, and Junge Zhang. EPO: Explicit policy optimization for strategic reasoning in LLMs via reinforcement learning. In *Proceedings of ACL*, pp. 15371–15396, 2025.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. G-Eval: NLG evaluation using GPT-4 with better human alignment. In *Proceedings of EMNLP*, 2023.

Jianqiao Lu, Zhiyang Dou, Hongru Wang, Zeyu Cao, Jianbo Dai, Yingjia Wan, Yunlong Feng, and Zhijiang Guo. AutoPSV: Automated process-supervised verifier. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Alexander Meulemans, Simon Schug, Seijin Kobayashi, Nathaniel D. Daw, and Gregory Wayne. Would I have gotten that reward? Long-Term credit assignment by counterfactual contribution analysis. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Long Ouyang, Jeffrey Wu, Xu Jiang, et al. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Zehan Qi, Xiao Liu, Iat Long Iong, Hanyu Lai, Xueqiao Sun, Jiadai Sun, Xinyue Yang, Yu Yang, Shuntian Yao, Wei Xu, Jie Tang, and Yuxiao Dong. WebRL: Training LLM web agents via self-evolving online curriculum reinforcement learning. In *International Conference on Learning Representations (ICLR)*, 2025.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Amrith Setlur, Chirag Nagpal, Adam Fisch, Xinyang Geng, Jacob Eisenstein, Rishabh Agarwal, Alekh Agarwal, Jonathan Berant, and Aviral Kumar. Rewarding progress: Scaling automated process verifiers for LLM reasoning. In *International Conference on Learning Representations (ICLR)*, 2025.

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

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel M. Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F. Christiano. Learning to summarize from human feedback. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Minzheng Wang, Yongbin Li, Haobo Wang, Xinghua Zhang, Nan Xu, Bingli Wu, Fei Huang, Haiyang Yu, and Wenji Mao. Adaptive social learning via mode policy optimization for language agents. In *International Conference on Learning Representations (ICLR)*, 2026.

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. Math-shepherd: Verify and reinforce LLMs step-by-step without human annotations. In *Proceedings of ACL*, 2024a.

Ruiyi Wang, Haofei Yu, Wenxin Zhang, Zhengyang Qi, Maarten Sap, Graham Neubig, Yonatan Bisk, and Hao Zhu. SOTOPIA-$\pi$: Interactive Learning of Socially Intelligent Language Agents. *arXiv preprint arXiv:2403.08715*, 2024b.

Zhilin Wang, Yi Dong, Olivier Delalleau, Jiaqi Zeng, Gerald Shen, Daniel Egert, Jimmy J. Zhang, Makesh Narsimhan Sreedhar, and Oleksii Kuchaiev. HelpSteer2: Open-source dataset for training top-performing reward models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024c.

Xiyu Wei, Qingwei Zong, Xiaoguang Li, Eugene J. Yu, and Sujian Li. QuRL: Rubrics as judge for open-ended question answering. In *International Conference on Learning Representations (ICLR)*, 2026.

Zeqiu Wu, Yushi Hu, Weijia Shi, Nouha Dziri, Alane Suhr, Prithviraj Ammanabrolu, Noah A. Smith, Mari Ostendorf, and Hannaneh Hajishirzi. Fine-grained human feedback gives better rewards for language model training. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Tian Xia, Zhiwei He, Tong Ren, Yibo Miao, Zhuosheng Zhang, Yang Yang, and Rui Wang. Measuring bargaining abilities of LLMs: A benchmark and a buyer-enhancement method. In *Findings of the Association for Computational Linguistics: ACL 2024*, 2024.

Weiran Yao, Shelby Heinecke, Juan Carlos Niebles, Zhiwei Liu, Yihao Feng, Le Xue, Rithesh Murthy, Zeyuan Chen, Jianguo Zhang, Devansh Arpit, Ran Xu, Phil Mui, Huan Wang, Caiming Xiong, and Silvio Savarese. Retroformer: Retrospective large language agents with policy gradient optimization. In *International Conference on Learning Representations (ICLR)*, 2024.

Haofei Yu, Zhengyang Qi, Yining Zhao, Kolby Nottingham, Keyang Xuan, Bodhisattwa Prasad Majumder, Hao Zhu, Paul Pu Liang, and Jiaxuan You. SOTOPIA-RL: Reward design for social intelligence. *arXiv preprint arXiv:2508.03905*, 2025.

Hongyi Yuan, Zheng Yuan, Chuanqi Tan, Wei Wang, Songfang Huang, and Fei Huang. RRHF: Rank responses to align language models with human feedback. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Yifu Yuan, Jianye Hao, Yi Ma, Zibin Dong, Hebin Liang, Jinyi Liu, Zhixin Feng, Kai Zhao, and Yan Zheng. Uni-RLHF: Universal platform and benchmark suite for reinforcement learning with diverse human feedback. In *International Conference on Learning Representations (ICLR)*, 2024.

Thomas Zeng, Shuibai Zhang, Shutong Wu, Christian Classen, Daewon Chae, Ethan Ewer, Minjae Lee, Heeju Kim, Wonjun Kang, Jackson Kunde, Ying Fan, Jungtaek Kim, Hyung Il Koo, Kannan Ramchandran, Dimitris Papailiopoulos, and Kangwook Lee. VersaPRM: Multi-domain process reward model via synthetic reasoning data. In *International Conference on Machine Learning (ICML)*, 2025.

Shaokun Zhang, Ming Yin, Jieyu Zhang, Jiale Liu, Zhiguang Han, Jingyang Zhang, Beibin Li, Chi Wang, Huazheng Wang, Yiran Chen, and Qingyun Wu. Which agent causes task failures and when? On automated failure attribution of LLM multi-agent systems. In *International Conference on Machine Learning (ICML)*, 2025a.

Wenyuan Zhang, Tianyun Liu, Mengxiao Song, Xiaodong Li, and Tingwen Liu. SOTOPIA-Ω: Dynamic strategy injection learning and social instruction following evaluation for social agents. In *Proceedings of ACL*, pp. 24669–24697, 2025b.

Chujie Zheng, Zhenru Zhang, Beichen Zhang, Runji Lin, Keming Lu, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. ProcessBench: Identifying process errors in mathematical reasoning. In *Proceedings of ACL*, 2025.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging LLM-as-a-judge with MT-Bench and Chatbot Arena. In *Advances in Neural Information Processing Systems (NeurIPS), Datasets and Benchmarks Track*, 2023.

Xuhui Zhou, Hao Zhu, Leena Mathur, Ruohong Zhang, Zhengyang Qi, Haofei Yu, Louis-Philippe Morency, Yonatan Bisk, Daniel Fried, Graham Neubig, and Maarten Sap. SOTOPIA: Interactive evaluation for social intelligence in language agents. In *International Conference on Learning Representations (ICLR)*, 2024a.

Yifei Zhou, Andrea Zanette, Jiayi Pan, Sergey Levine, and Aviral Kumar. ArCHer: Training language model agents via hierarchical multi-turn RL. In *International Conference on Machine Learning (ICML)*, 2024b.

# A VALIDATED CAPABILITY-SPACE CONSTRUCTION FOR SOTOPIA IN REWARDWEAVER

The SOTOPIA capability space contains 40 validated Rubrics organized into seven behavioral families. We first summarize the structure of the space (Table A1), then present representative semantic entries (Table A2), and finally show how a frozen Rubric is operationalized into an executable evaluator (Section A.3).

Table A1: **Behavioral families in the SOTOPIA capability Rubric library.** The validated space contains 40 individual Rubrics across seven families.

| Family | # Rubrics | Behavioral Scope |
|---|---|---|
| Goal Expression and Persistence | 5 | Goal articulation and retention |
| Information Acquisition | 8 | Information elicitation and clarification |
| Information Use | 5 | Evidence use and adaptation |
| Concern Handling and Persuasion | 6 | Objection handling and negotiation |
| Goal Progression and Closure | 7 | Progress, strategy, and completion |
| Relationship Management | 5 | Empathy and interaction quality |
| Boundaries and Safety | 4 | Privacy, truthfulness, and social boundaries |
| **Total** | **40** | |

## A.1 CAPABILITY-SPACE OVERVIEW

Table A1 summarizes the seven behavioral families spanned by the 40-dimensional SOTOPIA capability space. The grouping is organizational: failure attribution and reward adaptation operate over individual Rubrics rather than family-level scores.

## A.2 REPRESENTATIVE CAPABILITY RUBRICS

Table A2 presents representative entries covering the major regions of the capability space. Definitions are condensed for readability while preserving their operative behavioral distinctions. Positive and negative evidence illustrate the intended semantic boundary of each Rubric.

## A.3 RUBRIC-SPECIFIC EVALUATOR OPERATIONALIZATION

We use USE OF KEY INFORMATION, introduced in Table A2, as a concrete example of Rubric-Specific Evaluator construction. The Rubric semantics are fixed before evaluator development; operationalization specifies when the Rubric applies, how it is scored, and how uncertainty is handled.

Table A3 summarizes the resulting decision structure.

Table A2: **Representative entries from the 40-dimensional SOTOPIA capability Rubric library.** The selected entries span goal expression, information acquisition and use, adaptation, persuasion, goal progression, relationship management, and behavioral boundaries. USE OF KEY INFORMATION is further operationalized in Section A.3.

| ID | Capability | Definition | Positive Evidence | Negative Evidence |
|---|---|---|---|---|
| `explicit-goal` | Explicit Goal Expression | Explicitly state a task-required request, position, information, feeling, boundary, or correction that has not yet been expressed. | Asks a phone-using companion to put the phone away and engage with the group. | Comments on the pleasant gathering without stating the underlying request. |
| `elicit-key-info` | Targeted Information Elicitation | Directly ask for task-relevant information that is required but not yet available. | Asks the landlord about the deposit policy for early lease termination. | Continues discussing property maintenance without asking about the needed policy. |
| `use-key-information` | Use of Key Information | Accurately use at least one known, relevant, and safely usable fact to support the current response. | Explains a fear of enclosed spaces when declining an escape-room proposal. | Argues only that the alternative is more popular, leaving the relevant constraint unused. |
| `adapt-to-new-info` | Adapt to New Information | Adjust a request or plan when newly disclosed information changes its feasibility. | Suggests Thursday after learning that the partner will be away on Wednesday. | Repeats the Wednesday request despite the disclosed trip. |
| `address-concern` | Address Concerns with New Options | After the partner raises a specific concern, propose an actionable adjustment that directly reduces that concern. | Proposes a small low-sugar cake with ventilation after concerns about smell and diet. | Repeats that baking is enjoyable without modifying the proposal. |
| `strategy-pivot` | Switch Strategies | After repeated rejection or deferral, use a new argument, angle, or plan rather than repeating the same request. | Proposes a three-month hybrid-work trial after repeated remote-work rejections. | Repeats the same remote-work request without changing the argument or plan. |
| `empathic-acknowledgment` | Respond with Empathy | Acknowledge the partner's expressed negative emotion or difficulty before continuing the task interaction. | Recognizes a colleague's exhaustion before proposing how to divide the remaining report work. | Ignores the exhaustion and immediately demands additional overtime. |
| `secret-keeping` | Confidentiality Preservation | Avoid disclosing information explicitly marked as confidential when the related topic arises. | Discusses an investment proposal without revealing confidential financial circumstances. | Explicitly reveals information marked confidential in the task context. |
| `reject-improper-request` | Rejection of Improper Requests | Explicitly reject a request that violates social norms, law, or the agent's stated boundaries. | Refuses to make a false statement to cover for a roommate. | Agrees to falsely take responsibility for the roommate's rule violation. |
| `privacy-boundary` | Privacy Boundary Respect | Avoid probing task-irrelevant sensitive personal information when such information enters the conversation. | Offers to reschedule a contract discussion after a family hospitalization is mentioned. | Presses for details about the illness, costs, and unrelated financial circumstances. |

Table A3: **Operationalization of USE OF KEY INFORMATION.** Applicability is determined before scoring, and insufficient evidence is distinguished from genuine non-applicability.

| Field | Operationalization |
|---|---|
| Rubric ID | `fixed-use-key-information-v1` |
| Applicability | The Rubric applies when at least one safe, directly relevant fact, preference, constraint, or confirmed arrangement is available in the visible context before the target response. |
| Score $= 1$ | The target response accurately uses at least one currently relevant item of known information to explain, request, confirm, adapt, or respond. |
| Score $= 0$ | The Rubric applies, but the target response uses none of the relevant known information, or misstates a relevant fact, quantity, person, relation, or constraint. |
| Non-applicable ($N$) | No safe and directly relevant information is available for the current response. Failure to use available information is scored as $0$, not $N$. |
| Abstain ($U$) | The visible evidence is insufficient to determine applicability or scoring without guessing. |
| Boundary / Exclusion | Generic acknowledgment does not count as information use unless it clearly refers to a specific fact established in context. Earlier use of a fact does not imply that the current response uses it, and external knowledge is not introduced to supplement missing evidence. |

**Evaluator Operationalization.** Each candidate is operationalized by a dedicated evaluator and subjected to source-disjoint acceptance testing: the evaluator is developed on a development partition, and admission is decided on source-disjoint held-out cases. Rubric semantics remain fixed throughout evaluator development; prompt refinement only clarifies applicability, scoring boundaries, and abstention behavior. Once admitted, both the Rubric definition and evaluator are frozen across training stages. This separation between semantic definition and evaluator operationalization ensures that reward adaptation changes which capabilities receive training credit rather than what those capabilities mean.

# B SALES BENCHMARK CONSTRUCTION AND EVALUATION

We construct the Sales Benchmark to evaluate long-horizon interaction under structured business rules, asymmetric information, and conflicting objectives. The benchmark contains 400 health-service sales tasks across 20 scenario families, with 20 profile-conditioned instances per family. Among them, 350 are standard tasks and 50 are constraint-conflict tasks. This appendix details benchmark construction, quality control, data splits, evaluation, and representative tasks.

## B.1 CONSTRUCTION PROTOCOL

Each scenario family is instantiated with 20 buyer profiles. The 13 retained families use a bank of 20 buyer profiles, while the seven replacement families use a separate bank of 20 enterprise-buyer profiles, yielding 260 and 140 tasks, respectively.

**Model-assisted authoring.** We used GPT-5.6 Sol via Codex to assist with task drafting and refinement, conditioned on scenario specifications and buyer profiles. Construction includes explicit specification and revision of business rules, role goals, and private constraints rather than surface-level paraphrasing. The resulting task specifications are programmatically compiled into runtime views containing a shared scenario and role-specific information.

Prices, procedures, and service policies are fictional environment facts. The benchmark is entirely synthetic and is intended to evaluate interactive decision making and constraint following, not medical reasoning, diagnosis, or real-world healthcare recommendations; no real patient data are used. Each participant receives only its own role view; the counterpart's private information and internal authoring answer keys are excluded.

## B.2 AUTHORING TEMPLATE AND QUALITY CONTROL

The following compact template summarizes the constraints used to specify and revise tasks. It captures the construction protocol rather than reproducing every historical authoring request verbatim.

```
Construct or revise a health-service sales task from:
{scenario specification}, {buyer profile}, {business rules},
and {standard / constraint-conflict designation}.


Return:  a shared scenario, seller goal and private information,
and buyer goal and private information.


Specify decision-relevant options, prices, timing, and authority.
Preserve supplied facts and intended goal conflicts; flag missing
or inconsistent constraints for revision.  Keep private facts in
the appropriate role view and internal answer keys outside both
views.  Define outcomes observable within the interaction, without
assuming future third-party consent or actions.  Do not prescribe
an ideal dialogue or make unsupported medical claims.
```

**Quality control and deduplication.** Tasks are checked for schema completeness, sufficient decision-relevant information, internal consistency of business rules, appropriate role visibility, and observable completion conditions. Underspecified or inconsistent drafts are revised while preserving intentional goal conflicts.

Automated validation additionally checks for internal answer-key fields in runtime inputs and exact matches of private-information strings in unauthorized role views. Task-ID uniqueness and exact-match checks over serialized environment specifications verify that the final collection contains 400 distinct task records.

### B.3 SCENARIO FAMILIES

Table B1 summarizes the 20 scenario families. The displayed names are abbreviated; complete task specifications and identifiers will accompany the released benchmark.

Table B1: **Scenario families in the Sales Benchmark.**

| Family | Scenario |
|---|---|
| R01 | Staged corporate procurement |
| R02 | Data-use purpose separation |
| R03 | Cross-city service-network due diligence |
| R04 | Multi-site acceptance and objection terms |
| R05 | Corporate lead qualification |
| R06 | Sign-language and bilingual support |
| R07 | Contract, payment, and invoice entities |
| S01 | Budget-constrained add-ons |
| S02 | Final price confirmation |
| S04 | Retention against a cheaper competitor |
| S05 | Package reduction |
| S06 | Deposits and slot reservation |
| S10 | Time-constrained service selection |
| S13 | Paid report interpretation |
| S14 | Package-tier selection |
| S15 | Renewal timing |
| S16 | Referral rewards |
| S17 | Gift-package transfer |
| S19 | Follow-up examination conversion |
| S20 | Refund and replacement purchase |

### B.4 SPLIT AND EVALUATION

**Split and overlap checks.** We use a fixed 320/80 train–evaluation split, assigning 16 training and four evaluation tasks to each scenario family. The training split contains 287 standard and 33 constraint-conflict tasks; the evaluation split contains 63 standard and 17 constraint-conflict tasks.

No task IDs or exact serialized environment specifications overlap across the two partitions. Scenario families and profile banks are shared across the split, so the evaluation split measures generalization to unseen tasks within each scenario family. The 80-task evaluation set is reserved for final reporting; it never participates in reward adaptation or checkpoint selection.

**Outcome evaluation.** The outcome judge is Qwen2.5-72B. It receives the shared scenario, both participants' backgrounds and goals, and the complete dialogue, and applies the seven-dimensional SOTOPIA-EVAL rubric without additional sales-specific authoring answer keys. The reported Overall score is the arithmetic mean of the seven raw dimension scores.

This outcome evaluator is distinct from the process-reward evaluators used during training.

**Release.** We will release the benchmark alongside the paper, including the task specifications, fixed split manifest, construction instructions, and evaluation materials containing the actual scoring prompt, parser, and aggregation code.

### B.5 REPRESENTATIVE TASKS

We provide two representative tasks below. These are condensed English renderings of the underlying task specifications rather than complete runtime exports; personality fields and secondary business details are omitted for brevity.

**Standard: Refund and replacement purchase.** `sales-crafted-S20-P01`. A buyer requests a refund for an unused RMB 799 health-check package because an employer-provided examination duplicates several included items. The seller may offer fixed RMB 599 alternatives with a RMB

200 difference refund, but must preserve the buyer's unconditional full-refund option. The buyer privately knows which items are already covered and remains interested in a six-hormone panel. The interaction therefore requires eliciting existing coverage and preferences before seeking a voluntary replacement purchase.

**Constraint conflict: Gift-package transfer.** `sales-crafted-S17-P03`. The seller seeks a voluntary RMB 180 upgrade, whereas the buyer wants to transfer the gifted package without additional payment. Free transfer requires an independent recipient-confirmation link; the buyer cannot provide consent on the recipient's behalf. The seller must explain the free-transfer option and stop pressing the upgrade after an explicit refusal. Initiating a compliant free transfer does not satisfy the seller's paid-upgrade objective, and issuing the link does not by itself establish a completed transfer.

## C Additional Benchmark Results of RewardWeaver

Table C1 reports the full AmazonHistoryPrice results corresponding to Figure 3 in the main text. Table C2 presents the full results on the Sales Benchmark under Self-Play and Partner evaluation.

Table C1: **Main results on AmazonHistoryPrice.** All sellers are evaluated with `deepseek-v4-pro` as the buyer. The best result in each column is shown in **bold**.

| Method | MI Deal Rate ↑ | CI Correct No-Deal Rate ↑ | Task Success Rate ↑ | Normalized Seller Utility ↑ | Expected MI Seller Return ↑ |
|---|---|---|---|---|---|
| *Proprietary LLMs* | | | | | |
| `doubao-seed-2-0-pro` | 94.40 | **100.00** | 94.50 | 0.85 | 0.80 |
| `glm-5.3-flash` | **95.20** | **100.00** | **95.30** | 0.91 | 0.86 |
| `glm-5.2` | 92.00 | **100.00** | 92.20 | 0.93 | 0.86 |
| `qwen3.8-max` | 80.80 | **100.00** | 81.30 | 0.95 | 0.76 |
| `qwen3.8-flash` | 83.20 | **100.00** | 83.60 | 0.93 | 0.77 |
| `qwen3.7-plus` | 51.20 | **100.00** | 52.30 | 0.92 | 0.47 |
| `deepseek-v4-pro` | 83.20 | **100.00** | 83.60 | 0.87 | 0.73 |
| *Post-training on Qwen2.5-7B-Instruct* | | | | | |
| Qwen2.5-7B-Instruct w/ GRPO | 84.00 | 66.70 | 83.60 | 0.56 | 0.47 |
| **RewardWeaver (ours)** | **95.20** | **100.00** | **95.30** | **0.97** | **0.93** |

Table C2: **Main results on the Sales Benchmark.** In the Partner setting, the counterpart is Qwen2.5-72B-Instruct. The best result in each column is shown in **bold**.

| Method | Self-Play | | Partner | |
|---|---|---|---|---|
| | Sales Goal / Overall ↑ | Sales-Hard Goal / Overall ↑ | Sales Goal / Overall ↑ | Sales-Hard Goal / Overall ↑ |
| *Proprietary LLMs* | | | | |
| `doubao-seed-2-1-pro` | 8.55 / 4.36 | 7.29 / 3.99 | 8.81 / 4.43 | 7.47 / 4.10 |
| `qwen3.8-max` | 8.65 / 4.38 | 7.76 / 4.09 | 8.76 / 4.43 | 7.76 / 4.17 |
| `gemini-3.1-pro-preview` | 8.63 / 4.37 | 7.29 / 4.11 | 8.83 / 4.48 | 7.59 / 4.08 |
| `deepseek-v4-pro` | 8.68 / 4.30 | 7.24 / 3.97 | 8.67 / 4.39 | 7.82 / 4.17 |
| `glm-5.2` | 8.38 / 4.30 | 7.59 / 4.12 | 8.73 / 4.40 | **7.88** / 4.19 |
| *Post-training on Qwen2.5-7B-Instruct* | | | | |
| Qwen2.5-7B-Instruct | 8.33 / 4.30 | 7.18 / 3.97 | 8.41 / 4.33 | 7.12 / 3.99 |
| Qwen2.5-7B-Instruct w/ GRPO | 8.38 / 4.41 | 7.71 / 4.28 | 8.59 / 4.47 | 7.65 / **4.29** |
| **RewardWeaver (ours)** | **8.73 / 4.43** | **7.88 / 4.31** | **8.90 / 4.53** | 7.71 / 4.24 |

# D QUALITATIVE CASE STUDIES OF REWARDWEAVER

To complement the aggregate experiments, we examine how reward adaptation operates at the trajectory level through three representative cases from a complete SOTOPIA training run. Together, they illustrate three properties of the mechanism: stage-dependent bottleneck transition, contrastive validation of failure-derived Rubrics, and selective attribution under uncertainty.

## D.1 BOTTLENECK TRANSITION

Multiple low-outcome trajectories exhibit a recurrent pattern of verbose, abstract, and difficult-to-interpret language. In a representative right-of-way negotiation at a four-way stop, the agent receives Goal Completion $2/10$, and the task evaluator explicitly attributes the failure to unclear communication. The same pattern recurs across other trajectories in the cohort, indicating a stage-level bottleneck rather than an isolated error.

The subsequent attribution step identifies *Clear Expression* as a candidate capability. Following rubric generation and calibration, the next reward specification incorporates a criterion targeting the observed communication failures. In the following stage, repeated rollouts of the same scenario exhibit improved behavior, reaching Goal Completion scores of 7 and 9; the previous language degradation was not observed in the sampled follow-up rollouts. The failure distribution then shifts again: the new low-outcome cohort exhibits a different recurrent pattern, and the next adaptation step redirects emphasis toward *Concrete Solutions*.

The case illustrates the stage dependence of training relevance:

$$\text{recurrent failure} \rightarrow \text{bottleneck attribution} \rightarrow \text{reward adaptation} \rightarrow \text{new failure distribution}.$$

A capability need not remain a training priority once its associated bottleneck recedes, and a capability that was previously non-limiting may become dominant as the policy and interaction distribution evolve. Thus, REWARDWEAVER allocates process-level credit according to the current bottleneck rather than a fixed capability set.

As scorecards are regenerated between adaptation stages, the qualitative comparison is based on repeated evaluation of the same scenario.

## D.2 CONTRASTIVE VALIDATION OF FAILURE-DERIVED RUBRICS

A Rubric proposed from failure-side evidence is not necessarily specific to failure. A candidate evaluator for *Empathic Support* is calibrated on a matched set containing both high- and low-outcome trajectories. The evaluator is applicable to 23 trajectories, yet all 23 pass, yielding zero observed failures and very low score variation. It is flagged as collapsed and targeted for revision. The candidate rubric set fails the publication gate because its validity rate is below the required threshold. Rubric-set acceptance conditions govern publication, while Rubric-level diagnostics guide local revision.

This distinction is central:

$$\text{failure-derived} \not\Rightarrow \text{failure-specific}.$$

A Rubric may be motivated by failed trajectories yet fail to distinguish failure-relevant behavior on held-out contrastive examples. Such a signal provides little useful discrimination for process supervision. REWARDWEAVER therefore separates bottleneck discovery from reward-rubric calibration: failure evidence proposes candidate capabilities, while calibration determines whether the generated rubric set satisfies publication conditions and uses Rubric-level diagnostics to guide revision.

This mechanism differs from the Success–Failure Trajectory Attribution variant studied in RQ3, which introduces success trajectories during bottleneck attribution. Here, calibration occurs after rubric generation and governs publication of the candidate reward specification. Success trajectories are not required for the failure-attribution step; calibration remains required before publishing an updated reward specification.

### D.3 SELECTIVE ATTRIBUTION UNDER UNCERTAINTY

Failure-grounded attribution does not require every low-outcome trajectory to be mapped to an agent-side capability deficit. In one representative trajectory, the agent fails to achieve its task goal because the partner's private objective is to decline the interaction. The task evaluator explicitly attributes the outcome to the partner's preference, and no agent-side capability is activated.

Similar cases occur elsewhere in the run, including trajectories with low terminal outcomes despite all currently active capability checks being satisfied. These observations establish that terminal failure alone is insufficient evidence for capability-level credit assignment.

The attribution pipeline therefore admits three outcomes:

supported capability bottleneck, interaction-driven failure, insufficient evidence.

Capability-level training credit is assigned only when the failure admits an evidence-supported, policy-controlled bottleneck. This selectivity prevents partner behavior, scenario constraints, or ambiguous evidence from being silently converted into agent-side learning objectives.

Accordingly, failure-grounded attribution is not forced attribution: low outcome determines which trajectories warrant diagnosis, whereas capability-level credit is assigned only when the failure is supported by observable, policy-controlled evidence.

**Summary.** The three cases illustrate complementary aspects of adaptive process rewards. First, training relevance is stage-dependent and should track the current failure bottleneck. Second, failure-derived Rubrics require calibration before publication: rubric-set acceptance checks govern publication, while Rubric-level diagnostics guide revision. Third, attribution remains selective under uncertainty: low outcome is a trigger for diagnosis, not a guarantee of capability-level blame.

Together, the cases show that effective reward adaptation requires current relevance, discriminative criteria, and evidence-supported attribution; REWARDWEAVER therefore combines evidence-grounded candidate selection with rubric-set calibration.

# E Association Between Bottleneck Attribution and Subsequent Failure Reduction

We further analyze whether the bottleneck importance estimated by RewardWeaver is informative about subsequent changes in capability-level failure patterns.

## E.1 Analysis Setup

For each capability $d$ at training stage $k$, we define bottleneck importance as

$$I_{k,d} = F_{k,d}^{\text{fail}} - F_{k,d}^{\text{succ}},$$

where $F_{k,d}^{\text{fail}}$ is the fraction of low-outcome trajectories attributed to a deficiency in capability $d$, and $F_{k,d}^{\text{succ}}$ is the corresponding fraction among high-outcome trajectories. A larger $I_{k,d}$ therefore indicates that the capability deficiency is more strongly associated with low-outcome trajectories.

This contrast is constructed for analysis: the attribution mechanism itself (Section 2.3) operates on low-outcome trajectories only, and the success-side contrast is introduced here to characterize how informative the attribution ranking is.

We quantify the subsequent reduction of this failure mode as

$$\Delta F_{k,d} = F_{k,d}^{\text{fail}} - F_{k+1,d}^{\text{fail}},$$

such that a larger positive value corresponds to a larger decrease in the attributed failure rate in the following stage.

We evaluate this relationship over 240 capability–stage pairs spanning 40 capabilities and six adjacent training-stage transitions. We use Spearman rank correlation because the attribution mechanism primarily operates through relative capability priorities.

## E.2 Results

Table E1 summarizes the results. Bottleneck importance exhibits a positive association with next-stage failure reduction, with Spearman $\rho = 0.28$. In comparison, the current failure rate alone shows a weaker association with subsequent reduction ($\rho = 0.21$).

To determine whether the attribution signal provides information beyond the prevalence of the current failure mode, we additionally control for $F_{k,d}^{\text{fail}}$. The association remains positive, with partial Spearman $\rho = 0.22$. We further compare capabilities within the same training stage, obtaining a stronger association of $\rho = 0.31$.

Table E1: **Association between bottleneck attribution and subsequent failure reduction.** We measure the association between capability-level signals at stage $k$ and the reduction in attributed failures at stage $k + 1$.

| Analysis | Spearman $\rho$ |
|---|---|
| Bottleneck importance $I_{k,d}$ vs. $\Delta F_{k,d}$ | 0.28 |
| Current failure rate $F_{k,d}^{\text{fail}}$ vs. $\Delta F_{k,d}$ | 0.21 |
| $I_{k,d}$ vs. $\Delta F_{k,d}$, controlling for $F_{k,d}^{\text{fail}}$ | 0.22 |
| Within-stage association | 0.31 |

The comparison reveals two complementary properties of the attribution signal. First, its association is stronger than that of current failure prevalence alone. Second, the association remains positive after accounting for current failure prevalence and when capabilities are compared within the same training stage. Thus, the attribution signal provides a stage-dependent ranking whose higher values are associated with larger subsequent reductions in capability-level failure rates.

# F SENSITIVITY ANALYSIS OF REWARDWEAVER

We examine two design choices governing reward adaptation in REWARDWEAVER: the number of capabilities that receive concurrent training credit and the frequency with which this allocation is updated. Unless otherwise specified, the main configuration uses an active Rubric budget of $B = 6$ and updates reward allocation every 30 optimization steps.

## F.1 ACTIVE RUBRIC BUDGET

Table F1 varies the number of concurrently active Rubrics under both Self-Play and Partner evaluation. Performance is non-monotonic in $B$: neither reducing the active set nor activating more Rubrics consistently improves performance. The main setting $B = 6$ achieves the strongest results across all reported settings, suggesting that effective process supervision depends on concentrating training credit on a selective set of current bottlenecks rather than accumulating additional Rubrics.

Table F1: **Sensitivity to the active Rubric budget $B$.** Results are reported as Goal Completion / Overall under both Self-Play and Partner evaluation. The main configuration uses $B = 6$. Increasing the number of simultaneously active Rubrics does not monotonically improve performance, supporting selective rather than cumulative credit allocation.

| | Self-Play | | Partner | |
|---|---|---|---|---|
| $B$ | SOTOPIA | SOTOPIA-Hard | SOTOPIA | SOTOPIA-Hard |
| 2 | 8.46 / 4.32 | 7.50 / 3.97 | 8.38 / 4.17 | 7.46 / 3.92 |
| 4 | 8.31 / 4.29 | 7.79 / 4.16 | 8.21 / 4.19 | 7.37 / 4.03 |
| **6** | **8.72 / 4.47** | **7.90 / 4.38** | **8.49 / 4.47** | **7.66 / 4.22** |
| 8 | 8.49 / 4.44 | 7.67 / 4.25 | 8.31 / 4.25 | 7.59 / 4.15 |
| 10 | 8.50 / 4.46 | 7.63 / 4.36 | 8.34 / 4.37 | 7.57 / 4.21 |

## F.2 ADAPTATION INTERVAL

Table F2 varies the reward-adaptation interval under both Self-Play and Partner evaluation. Performance is again non-monotonic. The 30-step configuration provides the most consistently strong results across interaction modes and difficulty levels: it achieves the strongest Goal Completion scores under Self-Play and the strongest results across all Partner settings, while the 20-step setting is marginally higher on selected Self-Play Overall scores.

These results indicate that reward adaptation is sensitive to its update timescale rather than benefiting monotonically from either faster or slower updates.

Table F2: **Sensitivity to the reward-adaptation interval.** The main configuration updates reward allocation every 30 optimization steps. Results are reported as Goal Completion / Overall.

| | Self-Play | | Partner | |
|---|---|---|---|---|
| Interval | SOTOPIA | SOTOPIA-Hard | SOTOPIA | SOTOPIA-Hard |
| 10 | 8.28 / 4.41 | 7.73 / 4.36 | 8.32 / 4.33 | 7.59 / 4.20 |
| 20 | 8.50 / **4.47** | 7.81 / **4.39** | 8.27 / 4.19 | 7.47 / 4.03 |
| **30** | **8.72 / 4.47** | **7.90** / 4.38 | **8.49 / 4.47** | **7.66 / 4.22** |
| 60 | 8.21 / 4.26 | 7.44 / 3.95 | 8.34 / 4.28 | 7.49 / 4.04 |

Taken together, the two analyses highlight complementary requirements of adaptive process rewards: credit allocation should remain selective in capability space and operate at an appropriate adaptation timescale.

# G CAPABILITY RETENTION UNDER REWARD REALLOCATION

We further examine whether reallocating process-level credit causes previously emphasized capabilities to deteriorate once they leave the active reward set.

## G.1 EVALUATION PROTOCOL

We evaluate the policy at each stage on the same fixed diagnostic scenarios and score each tracked capability using a fixed evaluation criterion across all stages. For capability $d$ at stage $k$, we denote the resulting diagnostic score by $S_{k,d}$, computed as the mean normalized process score over applicable episodes.

Six capabilities receive process-level credit in the initial stage and are subsequently deactivated: *Address Concerns with New Options*, *State Benefits*, *Conditional Concessions*, *Maintain Goal Commitment*, *Regulate Pressure*, and *Switch Strategies*. *Adapt to New Information* is deactivated at a later stage. We additionally report three continuously active capability families, namely *Respond with Empathy*, *Early Disclosure*, and *Concrete Next Step*, as reference trajectories.

For a capability deactivated after stage $k$, we measure retention by its subsequent change $\Delta S_{t,d} = S_{k+t,d} - S_{k,d}$ relative to the final active stage.

## G.2 RETENTION RESULTS

Table G1 reports the capability trajectories. The six capabilities deactivated after the initial stage show no systematic regression: across 36 post-deactivation capability–stage observations, the mean change is $+0.10$, the median change is $+0.06$, and the minimum change is $-0.01$. No observation decreases by more than 0.02 from its final active score. *Adapt to New Information*, which is deactivated later, remains at 1.00 throughout all subsequent stages.

Table G1: **Capability retention across training stages.** † denotes capabilities deactivated after the initial stage, ‡ denotes a capability deactivated later, and unmarked dimensions remain active during the evaluated period.

| Capability | Stage 1 | Stage 2 | Stage 3 | Stage 4 | Stage 5 | Stage 6 | Stage 7 |
|---|---|---|---|---|---|---|---|
| Address Concerns with New Options† | 0.80 | 0.85 | 0.92 | 0.95 | 0.86 | 0.89 | 0.90 |
| State Benefits† | 0.84 | 0.88 | 0.97 | 0.98 | 0.95 | 0.94 | 0.98 |
| Conditional Concessions† | 0.85 | 0.90 | 0.98 | 0.93 | 0.86 | 0.90 | 0.96 |
| Maintain Goal Commitment† | 0.96 | 0.95 | 1.00 | 0.98 | 0.96 | 1.00 | 1.00 |
| Regulate Pressure† | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 |
| Switch Strategies† | 0.62 | 0.91 | 1.00 | 1.00 | 0.88 | 1.00 | 1.00 |
| Adapt to New Information‡ | 1.00 | 0.99 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 |
| Respond with Empathy | 0.99 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 |
| Early Disclosure | 0.92 | 0.90 | 0.94 | 0.94 | 0.91 | 0.92 | 0.99 |
| Concrete Next Step | 0.99 | 1.00 | 1.00 | 1.00 | 0.99 | 0.99 | 0.99 |

The event-aligned trajectories make the retention pattern particularly clear. Among the six early-deactivated dimensions, every post-deactivation observation remains close to or above its final active score. Several capabilities continue to improve despite receiving no further direct process-level credit: *Switch Strategies* rises from 0.62 to 1.00, *State Benefits* from 0.84 to 0.98, and *Address Concerns with New Options* from 0.80 to 0.90. The later-deactivated *Adapt to New Information* is similarly stable.

These results show that, on the fixed diagnostic set, shifting reward emphasis toward newly relevant capabilities does not systematically erase capabilities emphasized at earlier stages. The adaptive reward therefore behaves as a reallocation of training credit rather than requiring previously learned capabilities to be continually reinforced.