HERO: Hindsight-Enhanced Reflection from Environment Observations for Agentic Self-Distillation

Anonymous ACL submission

Abstract

Reinforcement learning typically improves multi-turn agent capabilities through the terminal outcome of the trajectories, which makes it difficult to determine credit assignments for each intermediate turns. Recent on-policy self-distillation methods offer a promising alternative by converting privileged feedback into dense token-level supervision through a self-teacher. Our study is motivated by the unexpected performance degradation observed when naively extending this paradigm to multi-turn settings, which we attribute to a lack of alignment between privileged feedback, such as successful trajectories or terminal outcomes, and the student's current decision context. We introduce HERO, a hindsight-enhanced self-distillation framework that uses next environment observations as locally aligned feedback. After each rollout, HERO reflects on the completed interaction to convert each observation into a compact turn-level diagnosis, that captures actionable feedback about the original action such as its necessity, validity or failure cause. On TauBench and WebShop, HERO improves task success and reduces unnecessary turns over environment-feedback-only self-distillation and GRPO. It is especially effective under limited training turn budgets, where successful rollouts are rare and GRPO provides weak reward-contrast signals.

1 Introduction

While Large Language Model (LLM) agents excel at multi-turn tool use, optimizing their interaction policies post-training remains challenging. When task outcomes can be scored by automatic verifiers or dedicated judge models, reinforcement learning (RL) methods such as PPO and GRPO provide a natural post-training framework (Schulman et al., 2017; Shao et al., 2024). However, these outcome-based methods often suffer from severe data-efficiency issues due to pathological credit assignment on the intermediate tool-call.

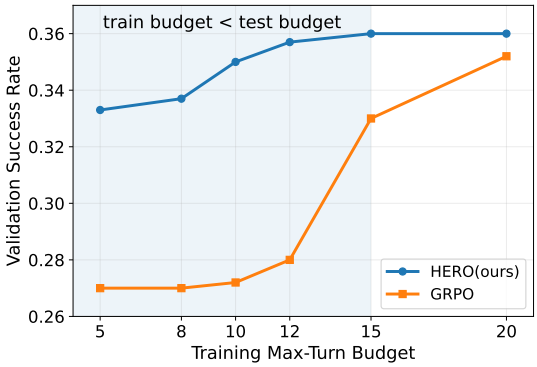

Figure 1: **HERO remains trainable under strict turn budgets.** Shaded region: training budget < test budget (15). GRPO receives weak reward variation when successful rollouts are rare; HERO still improves by learning from failed trajectories.

The issue is further amplified in long-horizon agent interactions, where successful rollouts are often rare because agents must reason over evolving observations and reach delayed outcomes under limited interaction budgets. As shown in Figure 1, we vary the training max-turn budget and evaluate the resulting validation success rate under the same test-time setting. When the interaction budget is small, most rollouts fail before reaching a successful terminal state, causing group-relative RL methods such as GRPO to receive little reward contrast across sampled trajectories. Consequently, the resulting advantages provide weak supervision signals that offer little guidance on local decisions. Moreover, even when an agent eventually succeeds, the outcome reward may still reinforce trajectories with redundant tool calls. These errors are easy to recognize in hindsight, but difficult to correct from scalar reward values.

A natural alternative is to convert outcome information into denser token-level supervision through on-policy self-distillation. Methods such as SDPO and OPSD condition a teacher on privileged feedback, such as task outcomes or successful demonstrations, and then distill the resulting distribution back into the original policy (Hübotter et al., 2026;

Zhao et al., 2026; Shenfeld et al., 2026). Such a strategy is shown to be especially effective on single-turn settings such as competitive coding where rich textual feedback is accessible (Zhang et al., 2026).

Interestingly, we find that a naive extension of SDPO degrades performance in multi-turn agent tasks. The key issue is that conditioning the teacher on a complete off-policy successful trajectory misaligns the teacher's privileged context with the student's local decision context. This observation suggests that the central question is not simply how to make feedback denser, but how to make it locally aligned with what the student actually observes and acts upon. *We argue that in multi-turn interaction, the most readily available local feedback signal is the next environment observation returned after each action.* Unlike terminal rewards or successful demonstrations, this observation immediately follows the student's action, and reflects how the environment responds to the agent's actual behavior. This motivates the central question of this work: how can we best leverage next observations as locally aligned feedback for multi-turn agent training?

In this paper, we propose HERO: Hindsight Enhanced Reflection from Environment Observations for agentic self-distillation. After each rollout, HERO uses the completed interaction to interpret the next observation at each turn and converts it into a compact turn-level diagnosis. This diagnosis summarizes the relevant hindsight evidence for the current decision, while avoiding direct conditioning on a complete off-policy successful trajectory. A self-teacher is then conditioned on the student's original decision context, the next environment observation, and the corresponding hindsight reflection, and re-evaluates the student's original action tokens. In this way, HERO turns naturally available interaction feedback into dense token-level supervision that is both informative and locally aligned with the student's on-policy behavior. Our contributions are:

- We identify that in multi-turn agent tasks, conditioning on off-policy successful trajectories creates a mismatch with the student's local decision and causes degraded performance.
- We argue that the most readily available local feedback signal is the next environment observation returned after each action.
- We propose HERO, an observation-grounded hindsight self-distillation framework that interprets next observations through compact turn-level reflections and conditions a self-teacher on the student's original decision context to produce dense token-level supervision.
- We empirically show that HERO improves success rate and execution efficiency on TauBench and WebShop, learns from failed or partial trajectories without successful demonstrations, and remains effective under limited training turn budgets where GRPO receives weak reward-contrast signals.

2 Preliminaries

2.1 Problem Formulation: Multi-turn Agentic RL

We consider a multi-turn tool-use environment. Given an initial task x, the agent interacts with the environment for at most T turns. At turn t, the policy observes the history

$$H_t = (x, y_1, o_1, \ldots, y_{t-1}, o_{t-1}),$$

where y_i is the assistant action, o_i is the environment observation. The student policy samples $y_t \sim \pi_\theta(\cdot \mid H_t)$. A complete trajectory is

$$\tau = (x, y_1, o_1, \ldots, y_T, o_T),$$

2.2 On-policy Self-Distillation

On-policy distillation trains a student on its own generated contexts (Agarwal et al., 2024; Lu and Lab, 2025). Given an input x, the student samples

$$y = (y_1, \ldots, y_T) \sim \pi_\theta(\cdot \mid x).$$

At decoding step t, the student distribution is

$$p_t = \pi_\theta(\cdot \mid x, y_{<t}).$$

A teacher provides a target distribution q_t, and the student is optimized to match it,

$$\mathcal{L}_{\text{OPD}}(\theta) = \mathbb{E}_{x, y \sim \pi_\theta}\left[\sum_{t=1}^{T} D(p_t, q_t)\right].$$

where $D(\cdot, \cdot)$ is a distributional distance, such as KL divergence or Jensen–Shannon divergence. Self-distillation removes the need for a separate expert teacher by instantiating the teacher from the same policy under a privileged context $c(x, y)$ (e.g.,

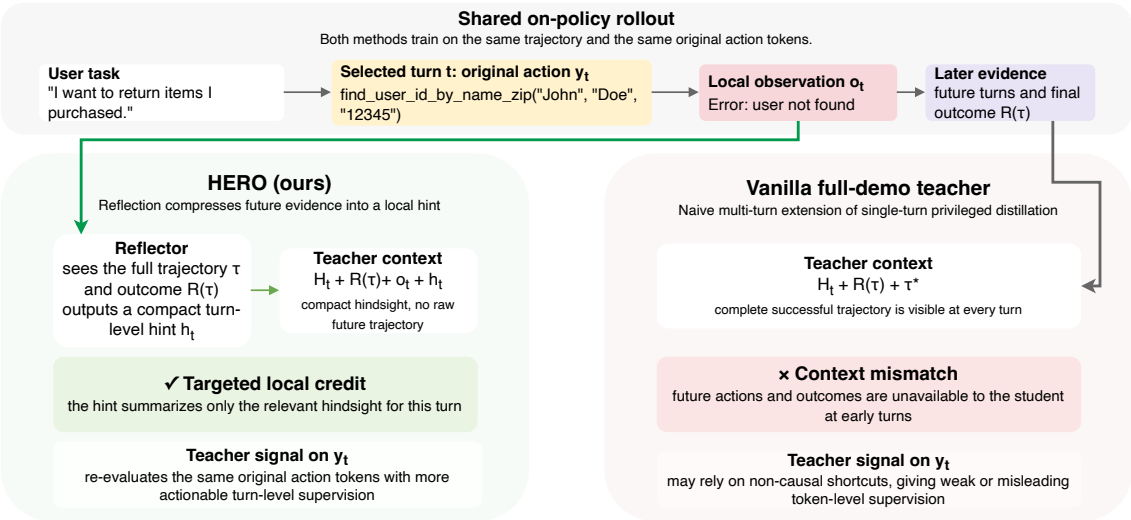

Figure 2: HERO compresses hindsight into turn-level privileged context. Full-demo self-distillation gives the teacher a complete successful trajectory at every turn, creating teacher–student context mismatch in early decisions. Environment-only feedback is local and often ambiguous. HERO first reflects over the completed rollout to extract a compact hint h_t, then conditions the self-teacher on H_t, o_t, $R(\tau)$, and h_t to produce targeted token-level supervision on the student's original action tokens.

environment feedback or a successful demonstration). The self-teacher distribution is

$$q_t = \mathrm{sg}\left[\pi_{\bar{\theta}}(\cdot \mid x, y_{<t}, c(x,y))\right],$$

where $\mathrm{sg}[\cdot]$ stops gradients through the teacher side, and $\bar{\theta}$ can be the current model, a delayed copy, or an EMA teacher. This converts privileged feedback into dense token-level supervision while keeping the student on-policy.

$$\mathcal{L}_{\text{Self-OPD}}(\theta) = \mathbb{E}_{x,y\sim\pi_\theta}\Big[\sum_{t=1}^{T} D\Big(\pi_\theta(\cdot \mid x, y_{<t}), \mathrm{sg}[\pi_{\bar{\theta}}(\cdot \mid x, y_{<t}, c(x,y))]\Big)\Big].$$

2.3 Jensen–Shannon Divergence

The symmetric Jensen-Shannon divergence is a commonly used distributional distance. Given the student distribution p_t and the self-teacher distribution q_t, define their mixture

$$m_t = \frac{1}{2}(p_t + q_t).$$

The Jensen–Shannon divergence is

$$D_{\mathrm{JSD}}(p_t, q_t) = \frac{1}{2}D_{\mathrm{KL}}(p_t \| m_t) + \frac{1}{2}D_{\mathrm{KL}}(q_t \| m_t),$$

where

$$D_{\mathrm{KL}}(p\|q) = \sum_{v\in\mathcal{V}} p(v)\log\frac{p(v)}{q(v)}.$$

3 Method

We propose HERO, **H**indsight-**E**nhanced **R**eflection from Environment **O**bservations, an observation-grounded self-distillation framework for multi-turn tool-use agents. HERO transforms each on-policy rollout into token-level supervision through three stages. First, the student policy generates a multi-turn trajectory under the standard causal interaction history. Second, after the rollout terminates, a reflector inspects the full trajectory and produces structured turn-level hints that diagnose local decision errors. Third, a self-teacher is conditioned on each turn-level hint and re-evaluates the student's original action tokens, producing dense distributional supervision for the student policy. Figure 2 gives an overview of HERO and contrasts it with the naive full-demo teacher.

3.1 Hindsight-Enhanced Reflection from Observations

After the rollout finishes, HERO invokes a reflector $\mathcal{R}$ that produces structured turn-level reflections:

$$h_{1:T} = \mathcal{R}(x, \tau, R(\tau)).$$

Each reflection is written abstractly as

$$h_t = (d_t, \hat{a}_t),$$

where d_t is a natural-language diagnosis and $\hat{a}_t$ is an optional corrected local action or target behavior.

User:
{{original_task_prompt}}
The following is feedback from your unsuccessful earlier attempt:
Trajectory outcome:
- trajectory_success: {{trajectory_success}}
- final_reward: {{final_reward}}
Environment feedback or tool observation:
{{environment_output}}
Reflection diagnosis: {{reflection_hint}}
Use the critique/hint to produce the corrected assistant action for the current scored turn.
Assistant: {{original_response}}

Figure 3: Teacher prompt template with trajectory outcome, environment feedback, and reflection diagnosis.

When the reflector finds no actionable signal for turn t, we set $h_t = \varnothing$. In our JSON implementation, the optional `confidence` field is treated only as textual metadata that helps the self-teacher interpret the reflection; it is not used as a numerical weight, filtering criterion, or optimization target.

Although $\mathcal{R}$ sees the full trajectory, the self-teacher later receives only the local hint h_t, not the raw future, so hindsight is compressed into a per-turn privileged signal.

3.2 Turn-level Privileged Self-Teacher

HERO uses the current policy in two roles. The *student* generates the original trajectory under the causal interaction history, while the *self-teacher* re-evaluates the same action tokens under a hindsight-augmented context.

A straightforward way to construct privileged context is to provide the teacher with the raw future trajectory after turn t,

$$\tau_{>t} = (y_{t+1}, o_{t+1}, \ldots, y_T, o_T).$$

The future trajectory contains delayed consequences of the current action. However, it provides an indirect credit-assignment signal. Since it can be long and noisy, the teacher may struggle to identify the earlier decision responsible for the eventual failure. It may also introduce a larger mismatch between the teacher context and the student's causal decision context. HERO therefore first converts the full trajectory into a compact turn-level hint h_t, which summarizes the relevant hindsight information for the current decision. For turn t, we build a hindsight-augmented teacher context

$$\widetilde{H}_t = \mathcal{P}\big(x,\, r_{<t},\, o_t,\, R(\tau),\, h_t\big),$$

where $\mathcal{P}$ is the fixed prompt template shown in Figure 3. Crucially, $\widetilde{H}_t$ contains only the local hint h_t, not the raw future trajectory. Let the assistant response at turn t be tokenized as $y_t = (y_{t,1}, \ldots, y_{t,|y_t|})$. At token position k, the student and (stop-gradient) self-teacher distributions are

$$p^{\text{stu}}_{t,k} = \pi_\theta(\cdot \mid H_t, y_{t,<k}),$$
$$p^{\text{tea}}_{t,k} = \text{sg}\big[\, \pi_{\bar{\theta}}(\cdot \mid \widetilde{H}_t, y_{t,<k}) \,\big].$$

We optimize the HERO objective by applying the self-distillation loss to the selected assistant turns. Let $\mathcal{I}(\tau)$ denote the set of assistant turns for which the reflection hint is successfully parsed and non-empty. Following the definition of D_{JSD} in §2.3, the objective is

$$\mathcal{L}_{\text{HERO}}(\theta) = \mathbb{E}_{\tau\sim\pi_\theta}\left[\frac{1}{|\mathcal{I}(\tau)|}\sum_{t\in\mathcal{I}(\tau)} \ell_t\right], \qquad (1)$$
$$\ell_t = \frac{1}{|y_t|}\sum_{k=1}^{|y_t|} D_{\text{JSD}}\left(p^{\text{stu}}_{t,k}, p^{\text{tea}}_{t,k}\right).$$

Remark: Learning from Failed Trajectories A key advantage of HERO is that it can learn even when no successful rollout is available. In GRPO-style training, the group-relative advantage is

$$A_i^{\text{GRPO}} = R(\tau_i) - \tfrac{1}{G}\textstyle\sum_{j=1}^{G} R(\tau_j).$$

When all G rollouts in a group fail, $R(\tau_i) = 0$ for every i and $A_i^{\text{GRPO}} = 0$, so GRPO receives no learning signal. In contrast, HERO can still obtain non-zero supervision through Eq. (1) whenever $h_t \neq \varnothing$ for some turn t. Failed trajectories therefore still teach the model to correct local execution errors — using concise tool-call formats, avoiding redundant API calls, or adhering to the required action schema.

4 Experiment

4.1 Experiment Setup

Datasets and Models. We evaluate HERO on multi-turn agentic tasks that require tool use and sequential decision making. Our main benchmarks are Tau-Bench Retail and WebShop (Yao et al., 2022a, 2024), covering structured API use for customer-service tasks and web-based multi-step shopping, respectively. We train on the retail domain and additionally evaluate on TauBench Airline as an out-of-distribution (OOD) domain to test cross-domain generalization. All methods are trained and evaluated on `Qwen3-4B-Instruct-2507` and

Table 1: Main results on multi-turn agent benchmarks. We report success rate (SR, ↑) and average number of turns / tool calls (Turns, ↓). TauBench-Airline is treated as an out-of-distribution benchmark.

Model	Method	TauBench-Retail		TauBench-Airline		WebShop	
		SR ↑	Turns ↓	SR ↑	Turns ↓	SR ↑	Turns ↓
Qwen3-4B-Instruct	Base	27.2	10.5	12.0	13.1	21.8	8.6
	Environment Feedback Only	30.4	11.5	12.5	13.7	37.5	10.4
	Full-Demo Privileged Teacher	22.5	12.4	8.5	13.9	17.4	10.6
	GRPO	33.3	13.4	18.0	14.0	65.2	**8.1**
	HERO (ours)	**34.7**	**9.6**	**19.5**	**10.1**	**68.9**	**8.1**
Qwen3-30B-A3B-Instruct	Base	33.6	11.9	30.0	12.3	32.7	8.5
	Environment Feedback Only	34.5	12.2	32.0	12.6	44.3	9.3
	Full-Demo Privileged Teacher	26.1	12.6	26.0	13.4	25.9	9.9
	GRPO	47.8	10.8	33.5	11.6	76.1	7.3
	HERO (ours)	**50.3**	**10.7**	**35.5**	**11.2**	**79.6**	**7.0**

`Qwen3-30B-A3B-Instruct-2507` under the same ReAct-style tool-use interface (Yang et al., 2025; Yao et al., 2022b).

Settings. For HERO, we use the generalized Jensen–Shannon divergence as the self-distillation objective with $\alpha = 0.5$. We apply a lower clipping threshold $\epsilon_{\min} = 0.4$ and an upper clipping threshold $\epsilon_{\max} = 10$. The training batch size is 16. Further implementation details, including prompt templates, reflection format, optimization hyperparameters, and decoding settings, are provided in Appendix A.

Baselines and metrics. We compare against four baselines: (i) **Base**, the instruction-tuned model without additional training; (ii) **Environment Feedback Only**, which conditions the self-teacher only on raw environment feedback or tool observations; (iii) **Full-Demo Privileged Teacher**, which naively extends single-turn privileged self-distillation to multi-turn agents by giving the self-teacher the complete successful trajectory or corrected demonstration at every turn; and (iv) **GRPO**, which optimizes scalar outcome rewards with group-relative advantages. The Full-Demo Privileged Teacher baseline is designed to test whether full future-trajectory exposure is a sufficient privileged signal, or whether it instead creates teacher–student context mismatch in early turns. We report mean@4 success rate and the average number of turns. Higher success and lower turn count indicate better and more efficient agents.

4.2 Main Results

Table 1 summarizes the main results across two model scales and three multi-turn agent benchmarks. Overall, HERO consistently improves success rate over the base model and the environment-feedback-only self-distillation baseline across all settings. Compared with GRPO, HERO achieves higher success rates in every benchmark while using comparable or fewer turns, suggesting that hindsight reflection improves not only task completion but also the efficiency of multi-turn execution.

For Qwen3-4B-Instruct, HERO improves over GRPO from 33.3% to 34.7% on TauBench-Retail, from 18.0% to 19.5% on the out-of-distribution TauBench-Airline benchmark, and from 65.2% to 68.9% on WebShop. The improvement is particularly notable in terms of execution length on TauBench, where HERO reduces the average number of turns from 13.4 to 9.6 on Retail and from 14.0 to 10.1 on Airline. This indicates that reflection-augmented supervision helps the agent avoid unnecessary or repeated tool calls, especially in long-horizon tool-use environments.

The gains are also consistent for Qwen3-30B-A3B-Instruct. HERO improves over GRPO by 2.5 points on TauBench-Retail, 2.0 point on TauBench-Airline, and 3.5 points on WebShop, while slightly reducing the average number of turns in all three benchmarks. These results suggest that stronger models can better benefit from hindsight reflection, likely because they generate more accurate diagnostic hints and follow reflection-conditioned teacher signals more reliably.

Why not directly use full successful trajectories? Existing self-distillation methods such as SDPO (Hübotter et al., 2026) and OPSD (Zhao et al., 2026) are primarily designed for single-turn settings, where the privileged context is naturally aligned with the current decision. In multi-turn agent tasks, however, there is no canonical way to extend such methods, because the teacher may

condition on information that is unavailable to the student at early turns. We therefore evaluate a natural but naive extension, where the teacher prompt is augmented with the complete successful trajectory or corrected demonstration for all turns. This variant performs worse than the base model / environment-feedback baseline. We attribute this degradation to teacher–student context mismatch. At early turns, the teacher has access to future actions and outcomes that are unavailable to the student. Its distribution may therefore depend on non-causal shortcuts rather than local decision evidence. This can spread credit over the full demonstration and produce misleading token-level supervision on the original action tokens.

4.3 Ablation on Teacher Prompt

We study which teacher-prompt components contribute to HERO. Table 2 compares raw environment feedback, reflection-only feedback, the full HERO prompt, and an external-reflector variant.

The results show that reflection provides more actionable supervision than raw environment feedback. Raw tool observations are often local and underspecified. They describe the immediate outcome of an action, but not always the reason the action was suboptimal. Hindsight reflection instead converts delayed consequences into direct turn-level corrections. This helps the self-teacher identify redundant, malformed, over-specified, or unsupported tool calls.

Table 2: Ablation on the self-teacher prompt. Reflection provides more actionable supervision than raw environment feedback and remains useful without successful demonstrations. We report success rate (↑) and average turns (↓).

Teacher Prompt Variant	Success ↑	Turns ↓
Raw env feedback only	30.4	11.5
Reflection only	33.9	10.1
HERO	34.7	9.6
Reflection with external reflector	35.7	9.4

We also compare self-reflection with an external reflector implemented using GPT-4o (Hurst et al., 2024). The external reflector yields slightly better performance, suggesting that higher-quality reflections can further improve HERO. However, the gap between self-reflection and GPT-4o reflection is modest. This supports our hypothesis that many bottlenecks in agentic tool-use tasks are not complex reasoning failures, but local execution and instruction-following errors, such as tool-call format, argument selection, and repeated action patterns. Although the policy may make these mistakes during rollout, it can often diagnose them in hindsight and convert them into useful self-distillation signals.

Notably, HERO remains effective even without successful demonstrations. This indicates that the method is not merely imitating successful rollouts; it can also learn from failed or partial trajectories by diagnosing local execution errors. This property is important under strict turn budgets, where successful trajectories are rare and outcome-reward RL receives little learning signal.

4.4 General Capability Retention

We further evaluate whether agent post-training degrades general capabilities. We test the final checkpoints on MMLU (Hendrycks et al., 2020), MMLU-Pro (Wang et al., 2024), and IFEval (Zhou et al., 2023), which measure broad knowledge, challenging reasoning, and instruction following. For IFEval, we report prompt-level strict accuracy in the main table, as it is the most conservative metric; full IFEval results are provided in Appendix D. As shown in Table 3, both GRPO and HERO show no noticeable degradation on these benchmarks. This suggests that on-policy agent training can improve tool-use behavior without broadly damaging general knowledge or instruction-following ability.

Table 3: General capability retention after TauBench-Retail training. Both GRPO and HERO show no noticeable degradation on MMLU, MMLU-Pro and IFEval, while HERO achieves stronger agentic performance and efficiency in Table 1.

Method	MMLU	MMLU-Pro	IFEval
Base	59.60	34.38	82.62
GRPO	59.61	34.25	82.19
HERO	59.54	34.35	82.80

4.5 Learning Beyond Sparse Outcome Rewards

HERO provides useful learning signal in regimes where outcome-reward RL is weak.

Strict turn budgets. GRPO relies on reward variation within a rollout group. Under small turn budgets, most rollouts fail with reward 0, causing group-relative advantages to collapse. HERO

does not require successful rollouts. Failed trajectories can still produce reflection hints, which provide non-zero token-level supervision. Therefore, HERO can improve even when GRPO receives little useful gradient signal.

Single-rollout learning. GRPO also requires multiple rollouts per task to estimate group-relative advantages. With $G = 1$, its advantage is identically zero. HERO remains well-defined because its supervision comes from a reflection-conditioned self-teacher rather than reward contrast. Figure 4 shows that HERO with $G = 1$ recovers most of the improvement of $G = 4$, despite higher variance. This reduces environment interaction cost, which is important for long trajectories, real APIs, or high-latency simulators.

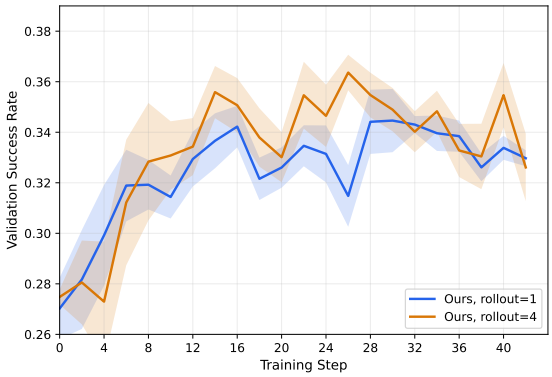

Figure 4: **HERO remains trainable with a single rollout per task.** Validation success rate on TauBench-Retail with $G = 4$ vs. $G = 1$; dashed line is base performance. GRPO is undefined when $G = 1$, since group-relative advantages collapse to zero.

Wall-clock efficiency. Although HERO introduces per-trajectory overhead from the reflection step and the teacher forward pass, it requires only $G = 1$ rollout per task while GRPO needs $G = 8$ to obtain non-zero group-relative advantages. The net effect is that HERO reaches comparable or better validation performance at lower total training cost; we report wall-clock curves in Appendix B.

4.6 Case Study: Reflection Sharpens Credit at the Forking Point

Figure 5 visualizes the actual per-token teacher JSD loss computed from a training trajectory, comparing the two teacher prompt variants on identical student tokens. The agent fabricates identity information, e.g., `John Doe` and `12345`, and calls `find_user_id_by_name_zip` instead of asking the user for real authentication information. The raw environment feedback only returns `user not found`, which is too ambiguous to identify the true error.

HERO resolves this ambiguity through hindsight reflection. The reflector diagnoses that the agent should not invent user information and should ask for the required name and zip code. This produces a stronger negative signal at the actual action forking point, especially on the incorrect tool call and the fabricated argument tokens. By contrast, the environment-feedback-only teacher mainly reacts to the local API failure, resulting in weaker and less targeted supervision.

5 Related Work

Multi-turn agentic RL. LLM agents solve interactive tasks by reasoning over histories and invoking external tools or APIs (Yao et al., 2022b; Schick et al., 2023; Patil et al., 2024). Benchmarks such as WebShop, tau-bench, and AppWorld evaluate these agents in multi-step environments with structured actions and delayed outcomes (Yao et al., 2022a, 2024; Trivedi et al., 2024). Outcome-based RL methods, including PPO and GRPO, optimize scalar rewards over complete trajectories (Schulman et al., 2017; Shao et al., 2024). However, such rewards provide weak credit assignment in multi-turn settings. Successful trajectories may still contain redundant or invalid intermediate actions. Failed trajectories often provide no usable gradient signal. Recent work addresses this issue with hierarchical value learning, structured advantages, or process reward models (Zhou et al., 2024; Zhang, 2026; Lightman et al., 2024). These methods usually require auxiliary critics or still reduce feedback to scalar signals. In contrast, HERO uses natural-language reflection to convert trajectory experience into turn-level privileged supervision.

On-policy distillation and self-distillation. Knowledge distillation trains a student to match a stronger teacher's distribution or representations (Kim and Rush, 2016; Sanh et al., 2019; Ko et al., 2024). However, standard distillation is often off-policy. The student is trained on fixed teacher- or dataset-generated contexts, but deployed on its own generations, leading to exposure bias and possible degradation during continued training (Agarwal et al., 2024; Xu et al., 2024; Chen et al., 2024). On-policy distillation mitigates this mismatch by supervising the student on its own sampled trajectories (Agarwal et al., 2024; Lu

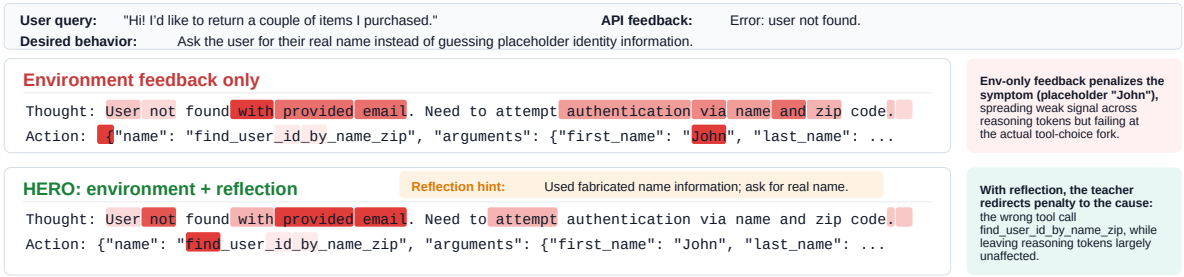

Figure 5: **HERO localizes credit to the wrong tool call.** Raw environment feedback only reports `user not found`, yielding weak or diffuse supervision. HERO converts the trajectory into a reflection hint, producing stronger negative signal at the tool-call forking point and on fabricated argument tokens. Red token backgrounds denote per-token teacher JSD loss, with darker red indicating a stronger negative signal.

and Lab, 2025; Yang et al., 2025; DeepSeek-AI, 2026). Because the objective is defined on the current policy distribution, it can better preserve or recover behaviors during midtraining and post-training (Chen et al., 2025; Shenfeld et al., 2026).

Self-distillation further removes the need for an external teacher by reusing the same model under privileged context (Snell et al., 2022). SDPO conditions the policy on environment feedback or successful rollouts and distills the feedback-informed distribution back into the original policy (Hübotter et al., 2026). Related work uses execution feedback, reference solutions, hints, or other privileged information for self-distillation (Ye et al., 2026; Zhao et al., 2026; Penaloza et al., 2026; Chen et al., 2025; He et al., 2026). We compare directly against the multi-turn extension of this family in Section 4.3.

Reflection and hindsight feedback. Natural-language reflection has been widely used to improve LLM behavior at inference time. Methods such as Reflexion, Self-Refine, and Self-Critique prompt models to critique and revise their own outputs (Shinn et al., 2023; Madaan et al., 2023; Saunders et al., 2022). TextGrad treats verbal feedback as a gradient-like signal for prompt optimization (Yuksekgonul et al., 2024), while other work uses pre- and post-reflection outputs to construct preference data (Stephan et al., 2024). These methods mainly use reflection to revise outputs, optimize prompts, or construct training targets.

HERO instead uses reflection as privileged context for self-distillation. The reflected diagnosis is not directly imitated as a target. It conditions a self-teacher, which produces token-level distributional supervision on the student's original action. This is related to Hindsight Experience Replay (Andrychowicz et al., 2017) and language-model extensions of hindsight learning (Liu et al., 2024; Zhang et al., 2023), but targets a different problem: assigning local credit to action tokens in multi-turn agent trajectories, rather than directly regenerating a corrected response.

6 Discussion and Conclusion

We introduced HERO, an observation-grounded hindsight self-distillation framework for multi-turn tool-use agents. Our starting point is that terminal outcome rewards and complete successful demonstrations are often poorly aligned with the student's local decision context in long-horizon interactions. Instead of conditioning the teacher on a complete off-policy trajectory, HERO addresses this by compressing hindsight evidence into compact turn-level diagnoses and using them, together with the student's original decision context, as privileged context for a self-teacher that produces dense token-level supervision.

Our experiments show that HERO improves task success and execution efficiency, especially when successful rollouts are rare and outcome-reward RL provides weak credit-assignment signals. These results suggest that effective self-distillation for multi-turn agents requires feedback that is not only dense, but also locally aligned with the student's decision context.

More broadly, HERO offers a middle ground between external-teacher distillation and pure outcome-reward RL. We see observation-grounded hindsight reflection as a promising direction for multi-turn agents, particularly when successful rollouts are rare, interaction costs are high, or raw feedback is ambiguous.

Limitations

HERO has several limitations. First, HERO relies on the model's own ability to reflect on its trajectories. This makes it most effective for agentic errors that are recognizable in hindsight, such as malformed tool calls, over-specified arguments, repeated API calls, or incorrect interaction patterns. These errors are common in tool-use agents, but the same mechanism may not be sufficient for tasks dominated by complex mathematical reasoning or deep multi-step problem solving, where the model may be unable to diagnose the underlying mistake by itself.

Second, HERO depends on the model's instruction-following and in-context learning abilities. The self-teacher must correctly interpret the reflection hint and translate it into token-level preference changes. As a result, HERO is better suited to instruction-tuned models than base models. In our experiments, larger and stronger instruct models benefit more from HERO, likely because they produce better reflections and follow hindsight-conditioned prompts more reliably.

References

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos Garea, Matthieu Geist, and Olivier Bachem. 2024. On-policy distillation of language models: Learning from self-generated mistakes. In *International Conference on Learning Representations*, volume 2024, pages 21246–21263.

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

Howard Chen, Noam Razin, Karthik Narasimhan, and Danqi Chen. 2025. Retaining by doing: The role of on-policy data in mitigating forgetting. *arXiv preprint arXiv:2510.18874*.

Zixiang Chen, Yihe Deng, Huizhuo Yuan, Kaixuan Ji, and Quanquan Gu. 2024. Self-play fine-tuning converts weak language models to strong language models. *arXiv preprint arXiv:2401.01335*.

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

Yinghui He, Simran Kaur, Adithya Bhaskar, Yongjin Yang, Jiarui Liu, Narutatsu Ri, Liam Fowl, Abhishek Panigrahi, Danqi Chen, and Sanjeev Arora. 2026. Self-distillation zero: Self-revision turns binary rewards into dense supervision. *arXiv preprint arXiv:2604.12002*.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2020. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, and 1 others. 2024. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*.

Jonas Hübotter, Frederike Lübeck, Lejs Behric, Anton Baumann, Marco Bagatella, Daniel Marta, Ido Hakimi, Idan Shenfeld, Thomas Kleine Buening, Carlos Guestrin, and Andreas Krause. 2026. Reinforcement learning via self-distillation. *arXiv preprint arXiv:2601.20802*.

Yoon Kim and Alexander M Rush. 2016. Sequence-level knowledge distillation. In *Proceedings of the 2016 conference on empirical methods in natural language processing*, pages 1317–1327.

Jongwoo Ko, Sungnyun Kim, Tianyi Chen, and Se-Young Yun. 2024. Distillm: Towards streamlined distillation for large language models. *arXiv preprint arXiv:2402.03898*.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2024. Let's verify step by step. In *International Conference on Learning Representations*, volume 2024, pages 39578–39601.

Hao Liu, Carmelo Sferrazza, and Pieter Abbeel. 2024. Chain of hindsight aligns language models with feedback. In *International Conference on Learning Representations*, volume 2024, pages 41395–41414.

Kevin Lu and Thinking Machines Lab. 2025. On-policy distillation. *Thinking Machines Lab: Connectionism*. Https://thinkingmachines.ai/blog/on-policy-distillation.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, and 1 others. 2023. Self-refine: Iterative refinement with self-feedback. *Advances in neural information processing systems*, 36:46534–46594.

Shishir G Patil, Tianjun Zhang, Xin Wang, and Joseph E Gonzalez. 2024. Gorilla: Large language model connected with massive apis. *Advances in Neural Information Processing Systems*, 37:126544–126565.

Emiliano Penaloza, Dheeraj Vattikonda, Nicolas Gontier, Alexandre Lacoste, Laurent Charlin, and Massimo Caccia. 2026. Privileged information distillation for language models. *arXiv preprint arXiv:2602.04942*.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

William Saunders, Catherine Yeh, Jeff Wu, Steven Bills, Long Ouyang, Jonathan Ward, and Jan Leike. 2022. Self-critiquing models for assisting human evaluators. *arXiv preprint arXiv:2206.05802*.

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

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

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, and 1 others. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

Idan Shenfeld, Mehul Damani, Jonas Hübotter, and Pulkit Agrawal. 2026. Self-distillation enables continual learning. *Preprint*, arXiv:2601.19897.

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

Charlie Snell, Dan Klein, and Ruiqi Zhong. 2022. Learning by distilling context. *arXiv preprint arXiv:2209.15189*.

Moritz Stephan, Alexander Khazatsky, Eric Mitchell, Annie S Chen, Sheryl Hsu, Archit Sharma, and Chelsea Finn. 2024. Rlvf: Learning from verbal feedback without overgeneralization. *arXiv preprint arXiv:2402.10893*.

Harsh Trivedi, Tushar Khot, Mareike Hartmann, Ruskin Manku, Vinty Dong, Edward Li, Shashank Gupta, Ashish Sabharwal, and Niranjan Balasubramanian. 2024. Appworld: A controllable world of apps and people for benchmarking interactive coding agents. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 16022–16076.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, and 1 others. 2024. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. *Advances in Neural Information Processing Systems*, 37:95266–95290.

Shusheng Xu, Wei Fu, Jiaxuan Gao, Wenjie Ye, Weilin Liu, Zhiyu Mei, Guangju Wang, Chao Yu, and Yi Wu. 2024. Is dpo superior to ppo for llm alignment? a comprehensive study. *arXiv preprint arXiv:2404.10719*.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

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

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

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

Tianzhu Ye, Li Dong, Xun Wu, Shaohan Huang, and Furu Wei. 2026. On-policy context distillation for language models. *arXiv preprint arXiv:2602.12275*.

Mert Yuksekgonul, Federico Bianchi, Joseph Boen, Sheng Liu, Zhi Huang, Carlos Guestrin, and James Zou. 2024. Textgrad: Automatic" differentiation" via text. *arXiv preprint arXiv:2406.07496*.

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

Tianjun Zhang, Fangchen Liu, Justin Wong, Pieter Abbeel, and Joseph E Gonzalez. 2023. The wisdom of hindsight makes language models better instruction followers. In *International Conference on Machine Learning*, pages 41414–41428. PMLR.

Yuwei Zhang, Sha Li, Changlong Yu, Qin Lu, Shuowei Jin, Chengyu Dong, Haoran Liu, Ilgee Hong, Xintong Li, Zhenyu Shi, and 1 others. 2026. Learning with rare success but rich feedback via reflection-enhanced self-distillation. *arXiv preprint arXiv:2605.12741*.

Siyan Zhao, Zhihui Xie, Mengchen Liu, Jing Huang, Guan Pang, Feiyu Chen, and Aditya Grover. 2026. Self-distilled reasoner: On-policy self-distillation for large language models. *arXiv preprint arXiv:2601.18734*.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. 2023. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*.

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

A Implementation Details

Training setup. We train all models for a single epoch over the training data. Rollouts are generated on-policy during training rather than reused from a fixed offline buffer. For HERO, we use a batch size of 16 prompts and a single rollout per prompt ($N = 1$) in all main experiments. The learning rate is 2×10^{-7}. For validation, we sample four responses per test problem and report mean@4 success rate or task score.

Hardware. We train Qwen3-4B-Instruct on 2×NVIDIA A100 (80GB) GPUs and Qwen3-30B-A3B-Instruct on 8×NVIDIA H200 GPUs. The same hardware configuration is used for all baselines (Environment Feedback Only, Full-Demo Privileged Teacher, GRPO) and HERO within each model scale, ensuring that wall-clock comparisons (Appendix B) are made under matched compute budgets.

Self-distillation objective. HERO uses the generalized Jensen–Shannon divergence with $\alpha = 0.5$ as the self-distillation loss. To reduce memory cost, we compute the distillation loss over the top-100 tokens under the student distribution at each position, with a tail bucket for the remaining probability mass. The loss is applied only to assistant-generated tokens and masked out on user prompts, tool observations, and environment feedback. We use an EMA self-teacher with update rate 0.001. For clipped importance weighting, we use a lower clipping threshold $\epsilon_{\min} = 0.4$ and an upper clipping threshold $\epsilon_{\max} = 10$. We set the maximum number of generated tokens per assistant turn to 512 and the maximum prompt length to 8192 tokens.

Reflection and teacher prompt. After each rollout, HERO constructs a reflection prompt from the original task, the full trajectory, tool observations, turn-level rewards, and final outcome. The reflector outputs structured turn-level hints, which are inserted into the teacher prompt for the corresponding assistant turn. Unless otherwise stated, the reflector is the policy itself prompted to reflect. In the external-reflector ablation, we use GPT-4o to generate the same structured reflection format. The teacher distribution is always computed by the policy or its EMA copy under the hindsight-conditioned prompt; the reflection itself is used only as privileged context and is not treated as a supervised target.

Learning Rate	Validation Success Rate
2×10^{-7}	32.4
5×10^{-7}	33.0
2×10^{-6}	**33.3**
5×10^{-6}	31.9

Table 4: Learning-rate sweep for GRPO on the TauBench-Retail validation split. We select 2×10^{-6} for the main experiments because it achieves the best validation success among the tested learning rates.

GRPO baseline. For GRPO, we use a rollout group size of 8 per prompt and batch size 16. We tune the learning rate separately for GRPO on the TauBench-Retail validation split. As shown in Table 4, 2×10^{-6} achieves the best validation success among the tested learning rates, and we therefore use it for the main GRPO baseline.

For HERO, we find that 2×10^{-7} gives stable optimization because the reflection-conditioned self-teacher provides dense token-level supervision. In contrast, GRPO receives a much sparser outcome-reward signal and learns slowly with the same learning rate. This choice favors a strong GRPO baseline rather than enforcing identical optimization hyperparameters across objectives with different gradient density and scale.

GRPO optimizes scalar outcome rewards with group-relative advantages and does not use self-distillation, hindsight reflection, or an EMA teacher. All baselines use the same ReAct-style interface, tool schema, environment, and evaluation protocol as HERO.

B Efficiency

Although HERO introduces additional computation from hindsight reflection and the teacher forward pass, it requires only a single rollout per prompt ($G = 1$). In contrast, GRPO requires a rollout group, and we use $G = 8$ in our main experiments, since group-relative advantages become ineffective when there is no reward variation within the group. Thus, GRPO spends most of its wall-clock time on generating multiple environment trajectories, while HERO trades substantially lower wall-clock time per training step for a cheaper reflection and self-teacher evaluation.

Figure 6 reports the per-step training time during TauBench-Retail training. Despite the extra reflection and distillation overhead, HERO is consistently faster than GRPO, taking roughly half the wall-clock time per training step. This is be-

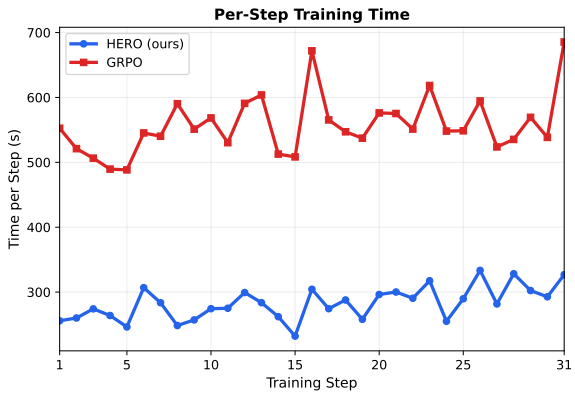

Figure 6: **Per-step wall-clock training time.** HERO uses a single rollout per prompt ($G = 1$) and remains faster than GRPO, which uses $G = 8$ rollouts per prompt to obtain group-relative advantages. Although HERO adds reflection and teacher-logit computation, the reduced rollout cost dominates the overhead.

cause HERO obtains a useful learning signal from each individual trajectory, including failed or partial ones, whereas GRPO must sample multiple rollouts per task to estimate group-relative advantages. As a result, HERO can achieve comparable or better validation performance with lower total training cost.

C Dataset Details

We summarize the datasets used in our experiments in Table 5. Our evaluation covers both structured API-based customer-service tasks and web-based shopping tasks. We train on TAUBENCH-RETAIL and WEBSHOP, and use TAUBENCH-AIRLINE only as an out-of-distribution transfer benchmark. For TauBench, task success is determined by environment-side database post-conditions, while WebShop provides a continuous reward based on how well the selected product matches the user instruction.

D Full IFEval Results

Table 6 reports the complete IFEval results under both strict and loose evaluation at the prompt and instruction levels. We observe no consistent degradation after agent post-training. Thus, we interpret these results as evidence of capability retention rather than improvement in general instruction following.

E Reflection Quality Analysis

To examine whether the reflector produces actionable supervision rather than plausible but incorrect critiques, we manually evaluate a sample of turn-level reflection hints.

Sampling protocol. We sample $N = 150$ turn-level hints from TauBench-Retail training rollouts, covering successful, failed, and partial trajectories that reach the turn budget. The sample contains 75 hints from the self-reflector and 75 hints from the GPT-4o reflector. For each sampled turn t, the annotator is shown the trajectory prefix H_t, the student action y_t, the subsequent observation o_t, and the reflection hint h_t, but is blind to the reflector identity.

Rubric. Each hint is labeled into one of four categories:

- **Correct**: the hint correctly identifies a real execution error or correctly confirms a sound action, and any suggested correction is consistent with the tool schema and task state.
- **Partially correct**: the hint identifies a real issue but misattributes the cause, assigns blame to an imperfect turn, or proposes a plausible but suboptimal correction.
- **Incorrect**: the hint is factually wrong, violates the tool schema, or references information unavailable at turn t.
- **Vacuous / non-actionable**: the hint is well-formed but provides little usable supervision, e.g., it marks a failed trajectory as correct or identifies an error without suggesting a repair.

Labels were assigned by one author following written guidelines. A second author independently labeled 30 hints, yielding Cohen's $\kappa = 0.87$.

Results. Table 7 reports the label distribution. The self-reflector produces 58.9% correct and 9.4% partially correct hints. The GPT-4o reflector improves the correct or partially correct rate from 68.3% to 75.6%, mainly by reducing incorrect hints. The remaining gap comes primarily from two failure modes: causality violations, where the self-reflector references future observations as if they were available at turn t, and multi-turn attribution errors, where it blames the most recent turn instead of the earlier forking point.

Benchmark	Train	Test	Description
TAUBENCH-RETAIL (Yao et al., 2024)	500	115	Multi-turn customer-service tasks with structured API calls over the retail domain, including order lookup, refunds, and exchanges. Task success is evaluated by binary rewards based on database post-conditions.
TAUBENCH-AIRLINE (Yao et al., 2024)	–	50	Out-of-distribution evaluation domain with a different tool schema and workflow, covering flight booking, cancellation, and seat changes. We use this split for evaluation only.
WEBSHOP (Yao et al., 2022a)	6219	691	Web-based multi-step shopping benchmark. The agent navigates product pages, selects attributes, and purchases an item matching a natural-language instruction.

Table 5: Dataset overview. "Train" and "Test" denote the number of unique task instances in each split. We train on TAUBENCH-RETAIL and WEBSHOP, and use TAUBENCH-AIRLINE only for out-of-distribution evaluation.

Table 6: Full IFEval results after agent training. We report both prompt-level and instruction-level accuracy under strict and loose evaluation.

Method	Inst. Loose	Inst. Strict	Prompt Loose	Prompt Strict
Base	89.93	88.13	85.40	82.62
GRPO	89.80	88.60	86.70	82.19
HERO	89.68	88.13	86.32	82.80

Label	Self-reflector	GPT-4o reflector
Correct	58.9%	60.3%
Partially correct	9.4%	15.3%
Incorrect	17.9%	12.0%
Vacuous / non-actionable	13.8%	12.4%

Table 7: Manual evaluation of reflection quality on 150 sampled turn-level hints.

Discussion. The results explain why self-reflection performs close to GPT-4o reflection in Table 2: most self-generated hints are already correct or at least point to a real issue. Stronger reflectors mainly improve the long tail of attribution and causality errors. In practice, even partially correct hints can provide useful learning signal because the self-teacher loss is computed at the token level; as shown in Figure 5, the loss can still concentrate on the wrong tool name or fabricated argument tokens rather than uniformly penalizing the whole trajectory.

Failure modes. We observe three recurring failure modes: (i) *hindsight leakage*, where the reflection uses information only available in later turns; (ii) *recency bias*, where the reflection attributes failure to the latest turn instead of the true earlier error; and (iii) *over-confident corrections*, where the hint proposes a plausible but unsupported next action.

E.1 Qualitative Examples of Reflection Hints

We further show representative examples from TauBench and WebShop. The examples illustrate both useful reflection behavior and common failure modes.

TauBench: Correct Hint

Outcome. The trajectory succeeds with sequence reward 1.0.
Hint. All assistant turns are diagnosed as `Correct`, with `next_action_target = null`.
Interpretation. The reflection is consistent with the successful trajectory and does not inject unnecessary corrections.

TauBench: Partially Correct Hint

Hint. The reflection says the agent identified an available white wireless-earbuds variant and price difference, but failed to obtain explicit user confirmation before proceeding.
Interpretation. The hint recognizes real task progress while also identifying a missing policy requirement. It gives useful corrective supervision, although the original trajectory is not entirely wrong.

TauBench: Incorrect Hint

Hint. The reflection incorrectly marks the current lookup action as reasonable:

```
find_user_id_by_name_zip(name, zip)
```

After observing `Error: user not found`, it recommends asking the user for more identity information, such as a phone number or email address, before calling the same type of lookup tool again.
Interpretation. This hint gives the wrong credit assignment. It treats the failed lookup as an information-collection problem, while the underlying issue is that the user has not placed any valid order. Therefore, additional identity queries would still fail to support the return request. The hint should instead identify the tool-use strategy as unhelpful and guide the agent to verify the existence of a valid purchase or order record.

TauBench: Vacuous / Non-actionable Hint

Outcome. The trajectory fails with sequence reward 0.0.
Hint. Every turn is labeled `Correct`, and every `next_action_target` is null.
Interpretation. The hint provides no explanation for the failed trajectory and no actionable correction, so it cannot provide useful distillation signal.

WebShop: Correct Hint

Outcome. The trajectory succeeds with sequence reward 1.0.
Hint. The reflection marks the search query, product click, attribute selection, and final purchase as aligned with the instruction.
Interpretation. The hint agrees with both the available actions and the final reward.

WebShop: Partially Correct Hint

Hint. The reflection says the selected product satisfies the price and product-type constraints, but the agent skipped verification of moisture-wicking and daily-wear requirements.
Interpretation. The hint captures a partial match: the selected item is plausible, but some important attributes remain unchecked.

WebShop: Incorrect Hint

Hint. The reflection proposes `click[yellow]` as the next action.
Context. At this turn, `click[yellow]` is not among the available actions.
Interpretation. The hint is invalid because WebShop actions must be selected from the current clickable set. Training on this hint would encourage an impossible action.

WebShop: Vacuous / Non-actionable Hint

Hint. The reflection correctly says that `buy now` was premature because the selected product did not satisfy the instruction, but gives `next_action_target = null`.
Interpretation. The hint identifies the error but does not provide a repair, making it weak supervision for correcting the failed trajectory.

F Reflection Prompt

Figure 7 shows the full reflection prompt used by HERO. After each on-policy rollout, the reflector $\mathcal{R}$ is invoked once on the full trajectory τ, the tool schema, and the final reward $R(\tau)$, and returns one structured hint h_t per assistant turn (Section 3.1). In all main experiments, $\mathcal{R}$ is the student policy itself (self-reflection); the ablation in Section 4 additionally evaluates an external LLM reflector (`gpt-4o`) to assess the effect of reflector strength.

Reflection Prompt

Role. You are an observation-grounded hindsight critic for a multi-turn tool-use agent. Given the original task, tool schema, full trajectory, and final reward, produce one structured JSON hint for each assistant turn.

Rules.

1. Reflect on every assistant turn, including correct turns.
2. Ground each diagnosis in the current action and its next environment observation. Use later trajectory evidence only to identify delayed consequences; do not expose future actions or observations verbatim.
3. Distinguish decision errors from tool-execution errors.
4. Fill `next_action_target` only when the corrected action is unambiguous; otherwise set it to `null` with `"medium"` or `"low"` confidence.
5. Avoid redundant actions, repeated confirmations, and re-asking already confirmed information.
6. Output only valid JSON, with no markdown or extra text.

Output format. Return a JSON object with one key per assistant turn: `"turn1"`, `"turn2"`, . . ., `"turnT"`.

```
{
  "turn1": {
    "diagnosis": "<brief causal diagnosis>",
    "confidence": "low",
    "next_action_target": null
  },
  "turn2": {
    "diagnosis": "<brief causal diagnosis>",
    "confidence": "high",
    "next_action_target": {
      "thought": "<one-line local plan>",
      "action": {
        "name": "<tool_name_or_respond>",
        "arguments": {"<arg_name>": "<arg_value>"}
      }
    }
  }
}
```

Figure 7: Reflection prompt used by HERO. The reflector produces one structured hint per assistant turn. Each hint is grounded in the current action and its next environment observation, while later trajectory evidence is used only to identify delayed consequences.