UOPD: UNCERTAINTY-AWARE INTERVENTION FOR ON-POLICY DISTILLATION OF MULTI-TURN AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

On-policy distillation (OPD) trains a student on its own rollouts using dense supervision from a teacher. In multi-turn environments, a mistake at a critical decision step can redirect the subsequent rollout toward poor outcomes. We use low teacher confidence on student actions to select high-uncertainty steps for correction. In a controlled ALFWorld study, a single teacher correction at a low-confidence step improves subsequent student behavior and task success, motivating selective intervention during distillation. We propose **UOPD**, an uncertainty-aware intervention method for on-policy distillation. At low-uncertainty turns, UOPD executes student actions and applies the standard OPD loss. At high-uncertainty turns, it samples and executes teacher actions and trains the student to imitate them through supervised fine-tuning, which minimizes forward Kullback–Leibler divergence in expectation. UOPD utilizes adaptive uncertainty thresholds to target a scheduled intervention rate. Empirically, we evaluate UOPD across a broad range of agentic tasks, including ALFWorld, WebShop, and Search, demonstrating its superior performance over OPD methods and their variants. UOPD improves WebShop score by up to 15.8% relative to standard OPD.

1 INTRODUCTION

Distilling a large teacher into a small student has become a standard stage of language-model post-training (Hinton et al., 2015; Agarwal et al., 2024). Among its variants, *on-policy distillation* (OPD) has emerged as the dominant recipe: rather than imitating a fixed corpus of teacher trajectories, the student samples its own rollouts and the teacher supplies a dense, token-level target at every state the student actually visits (Agarwal et al., 2024). This removes the exposure bias of supervised distillation and, by minimizing a reverse Kullback–Leibler (KL) under the student's own distribution, gives the student a mode-seeking target that it has the capacity to match. Recent work has extended OPD to multi-turn agents, which gather information, use tools, and act on successive observations to accomplish complex tasks (WANG et al., 2026; Zhong et al., 2026; Lu et al., 2026; Zhou et al., 2026a). Distilling these capabilities into smaller models is particularly valuable for efficient deployment, as completing a single task often requires many model calls.

In multi-turn settings, each action shapes subsequent observations and decisions. Student errors can therefore compound across turns, degrading trajectory quality and teacher supervision quality (WANG et al., 2026; Zhong et al., 2026; Zhu et al., 2026a). Standard OPD can reduce the probability of sampled poor actions through distillation, but does not directly provide better actions for the student to imitate. These poor actions are still executed during rollout collection and can lead to repeated mistakes and task failure. Existing approaches mitigate multi-turn drift by reweighting teacher supervision (Zhong et al., 2026; Zhou et al., 2026a), adapting rollout length (Yang et al., 2026b; WANG et al., 2026), or using teacher-generated prefixes (WANG et al., 2026; Liao et al., 2026). However, these approaches do not directly correct unreliable student actions at the turns where they arise. Executing such actions can lead to further mistakes and task failure, reducing the quality of training trajectories.

To address this gap, we consider *intervention*: replacing selected student actions before execution. As the teacher is generally more capable than the student, it can supply a replacement action for such an intervention, providing explicit corrective targets while guiding the subsequent rollout. We focus on deciding when such intervention is needed. The student may generate actions that elicit

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.

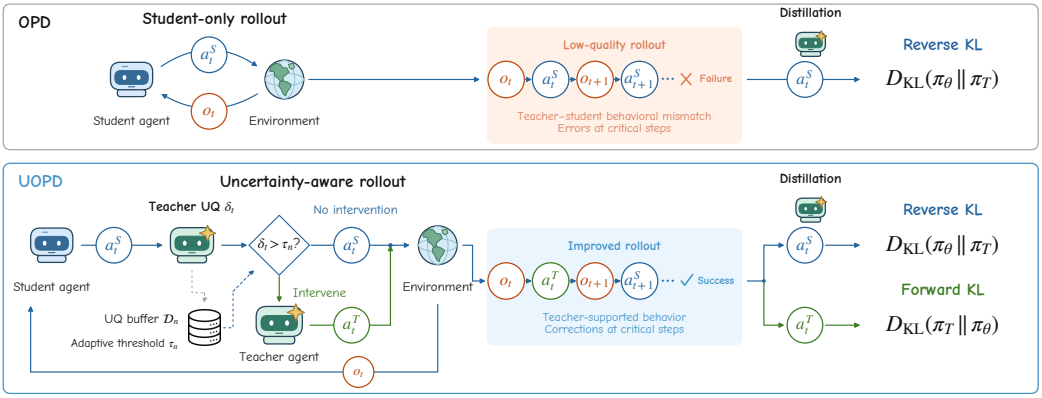


Figure 1: OPD versus UOPD. OPD executes student actions (a_t^S) given observations (o_t) and distills using reverse KL. UOPD uses teacher signals for uncertainty quantification (UQ) and an adaptive threshold to determine when the teacher intervenes. This decision jointly determines the executed action and the learning objective: reverse KL for student actions, or SFT on teacher actions (a_t^T). The latter minimizes forward KL in expectation.

high uncertainty from the teacher. This uncertainty offers a potential signal for identifying steps where a teacher correction could be useful. Our controlled study in Section 3 shows that a single teacher correction yields a larger improvement in task success when applied at a high-uncertainty turn than at a randomly selected turn. These results support using teacher uncertainty to select turns for intervention.

Motivated by these findings, we propose **Uncertainty-Aware Intervention for On-Policy Distillation (UOPD)** (Fig. 1). At each turn, the student proposes an action, and the teacher evaluates its uncertainty on that action. At low-uncertainty turns, UOPD executes the student action and trains with reverse KL. At high-uncertainty turns, it samples and executes a teacher action and trains the student to imitate it through supervised fine-tuning (SFT), which minimizes forward KL in expectation. Teacher intervention thus supplies an action that the student may rarely generate on its own and lets the student continue interacting from the resulting observation. UOPD uses uncertainty to decide when to learn from the student's own actions and when to introduce a teacher correction.

Our **contributions** are summarized as follows:

- We propose **UOPD**, which uses teacher uncertainty to guide action execution and supervision in multi-turn on-policy distillation. UOPD combines reverse-KL learning on student actions with SFT on executed teacher corrections, providing corrective targets and improving trajectory quality.
- We provide a theoretical analysis of the two roles of teacher corrections. We derive a rollout return improvement guarantee under average recoverability, and show that teacher-action imitation can improve student return even when reverse-KL supervision reduces it.
- Experiments on ALFWorld, WebShop, and multi-hop Search demonstrate that UOPD improves task performance over OPD and other state-of-the-art baselines across student model sizes while requiring fewer interaction turns, indicating the effectiveness and efficiency of our method.

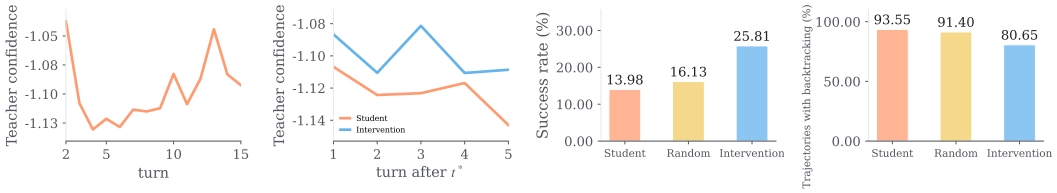
2 Related Work

On-policy distillation. On-policy distillation trains a student on its own generations while querying a teacher for dense token-level supervision (Agarwal et al., 2024). Recent work connects this objective to policy-gradient learning (Oh et al., 2026; Yang et al., 2026a), organizes distillation methods by their rollout source and divergence direction (Zhao et al., 2026), and studies how the choice of divergence affects policy improvement (Agrawal et al., 2026). Other analyses show that the effectiveness of OPD depends strongly on student–teacher support overlap and that teacher supervision can become unreliable on student-generated prefixes (Zhu et al., 2026a; Li et al., 2026c). Several methods address this issue at the token level by adapting the KL direction (Jia et al., 2026; Zhu et al., 2026b), selecting supported tokens (Wang et al., 2026), or mixing teacher and student prefixes (Xu et al., 2025b; Zhang et al., 2026a). Orthogonal to where supervision is applied, a further group of variants changes what

the teacher provides: rubrics or preference pairs induced from teacher text stand in for logits when the teacher is black-box (Fang et al., 2026; Singh et al., 2025), privileged context conditions the teacher's scores (Yu et al., 2026), and another line couples teacher supervision with an explicit task reward inside a single objective (Xu et al., 2025a; Zhang et al., 2026d; Xu et al., 2026; Zhang et al., 2026b), following earlier on-policy policy distillation for control (Spigler, 2025). Teacher uncertainty is also used to adapt token-level distillation objectives (Jin et al., 2026; Ke et al., 2026), while other work obtains corrective supervision by probing alternative continuations or backtracking to an earlier reasoning prefix (Qu et al., 2026; Wang et al., 2026). UOPD selects corrections during environment interaction, where executing a corrected action also changes the states visited next.

Multi-turn agent distillation and intervention. In multi-turn environments, errors alter subsequent observations and cause distribution shift to accumulate across turns (WANG et al., 2026; Zhong et al., 2026; Zhou et al., 2026a). Existing approaches mitigate this drift by reweighting or filtering supervision (Zhong et al., 2026; Yang et al., 2026b; Lu et al., 2026), limiting or adapting rollout exposure (WANG et al., 2026; Zhang et al., 2026c; Liang et al., 2026; Zhou et al., 2026b), or modifying the roll-in trajectory through teacher prefixes, refinement, or lookahead feedback (Liao et al., 2026; Jiang et al., 2026; Liu et al., 2026; Li et al., 2026b). SAGE-OPD (Zhou et al., 2026a) uses teacher judgments and confidence to scale turn-level supervision. Teacher actions also guide rollouts through scheduled turn mixing (Li et al., 2026a) or corrections selected and validated after rollout collection (Chen et al., 2026). UOPD uses teacher uncertainty to target critical steps and intervene before the student's proposed action is executed, providing a corrective learning target and redirecting the subsequent rollout. UOPD also connects to DAgger-style expert supervision at learner-visited states (Ross et al., 2011; Agrawal et al., 2026) and HG-DAgger's expert takeover (Kelly et al., 2019). It automates takeover using teacher uncertainty on student actions and trains the student on the executed corrections.

3 Why Is Intervention Needed Along the Student's Rollout?



(a) Student rollout **(b)** After intervention **(c)** Intervention benefit **(d)** Backtracking behavior

Figure 2: One teacher intervention at a low-confidence step improves rollout quality. (a) Teacher confidence fluctuates across turns of student-only rollouts. (b) Replacing one low-confidence action with a teacher action raises teacher confidence on subsequent actions generated by the student. (c)(d) This intervention increases task success and reduces backtracking compared with both student-only rollouts and random intervention. These results support using teacher uncertainty to select steps where a correction improves the rest of the rollout.

A poor student action can lead to further mistakes, affecting both task completion and the trajectory used for distillation. Standard OPD provides teacher supervision along this trajectory, but does not prevent the action from being executed. Correcting the action before execution can instead allow the student to continue from a better state and learn from the resulting trajectory. We therefore study whether teacher uncertainty identifies useful intervention points and whether a single teacher correction improves the student's subsequent rollout and task performance.

Experimental Setup. We conduct a controlled study on ALFWorld (Shridhar et al., 2021), keeping both models fixed. We use teacher confidence on student-generated actions to identify high-uncertainty steps. On the same tasks, we compare student-only rollouts, intervention at a low-confidence step, and intervention at a random step. Both intervention conditions replace exactly one student action with a teacher action, then return control to the student under the same turn limit. Appendix C provides the full protocol.

Observation 1: uncertainty varies across decision steps. Teacher confidence falls and recovers along student rollouts (Fig. 2(a)), showing that the teacher's support for student-generated actions changes within an interaction. Low-confidence actions are weakly supported by the teacher and

provide candidates for correction. In a multi-turn environment, executing such an action also determines the observations and decisions that follow, so its consequences can extend beyond the current turn. This motivates evaluating the student's proposed action in its current interaction history to decide when intervention may be useful.

Observation 2: one correction improves subsequent student behavior. After intervention, the teacher assigns higher confidence to subsequent student-generated actions (Fig. 2(b)). The teacher acts only once, and the student resumes without any parameter update: the benefit extends to actions the student generates on its own after the correction. Task success also rises from approximately 14% to 26%, compared with 16% for random intervention (Fig. 2(c)). The two intervention conditions use the same amount of teacher assistance, yet selecting a low-confidence step produces a larger gain. These results show that both the action correction and its timing matter for the subsequent rollout.

Observation 3: intervention helps correct student errors. Student errors can alter subsequent interaction and lead to behavior that fails to advance the task. We examine this effect in ALFWorld using returns to previously visited states as a task-specific behavioral indicator. Fig. 2(d) shows that low-confidence intervention reduces the fraction of trajectories that leave a state and later return to it, relative to both student-only rollouts and random intervention. Together with higher task success, this change suggests that a teacher correction can help the student recover from mistakes and make progress toward completing the task. A well-placed teacher correction can thus improve trajectory quality by changing how the student continues to act.

Takeaway. Teacher correction at a high-uncertainty step improves subsequent student behavior and task success. Its advantage over random intervention shows that correction timing matters, motivating teacher uncertainty as a signal for deciding when to intervene.

4 Uncertainty-Aware Intervention for On-Policy Distillation

4.1 Notation and Preliminaries

We consider a multi-turn interaction starting from a task input q. Before turn t, the agent conditions on the interaction history $h_t = (q, a_0, o_1, \ldots, a_{t-1}, o_t)$, with $h_0 = q$. A policy π samples an action response $a_t \sim \pi(\cdot \mid h_t)$. Executing the action in the environment produces an observation o_{t+1} and updates the history to $h_{t+1} = (h_t, a_t, o_{t+1})$. A trajectory of H turns is $\tau = (q, a_0, o_1, \ldots, a_{H-1}, o_H)$.

Let π_θ be the student policy and π_T a frozen teacher. In standard OPD, the student generates every action, $a_t \sim \pi_\theta(\cdot \mid h_t)$. The teacher provides dense token-level supervision on the resulting rollouts. The OPD objective minimizes the reverse Kullback–Leibler (KL) divergence from student to teacher at the visited histories:

$$\mathcal{L}_{\mathrm{OPD}}(\theta) = \mathbb{E}_{\tau \sim \pi_\theta}\left[\sum_{t=0}^{H-1} \mathrm{KL}(\pi_\theta(\cdot \mid h_t) \,\|\, \pi_T(\cdot \mid h_t))\right]. \tag{1}$$

For a sampled student action a_t, we write $\ell_t^{\mathrm{OPD}}(a_t; \theta) := \log \pi_\theta(a_t \mid h_t) - \log \pi_T(a_t \mid h_t)$. Its expectation over $a_t \sim \pi_\theta(\cdot \mid h_t)$ equals the KL divergence at h_t in Eq. 1.

Motivated by the observations in Section 3, UOPD selectively replaces student actions with teacher corrections during rollout collection.

4.2 UOPD: Uncertainty-Aware Intervention for On-Policy Distillation

UOPD consists of two components: an intervention rule that decides when to replace a student action with a teacher action, and a loss that specifies how the student learns at each turn.

Intervention Rule. Let $\mathcal{U}(\pi_T, \pi_\theta; h, a) \in \mathbb{R}$ be an uncertainty score for action a given history h, with larger values indicating greater uncertainty. The uncertainty score can be instantiated using teacher confidence or a student–teacher confidence gap; we provide the specific definitions in Appendix D.4.

Let n index batches of episodes collected during training, and let r_n be the target intervention rate for batch n. Given a buffer $\mathcal{D}_n$ of recent uncertainty scores of student actions, UOPD sets the intervention threshold to $\tau_n = \mathrm{Quantile}_{1-r_n}(\mathcal{D}_n)$. The target rate controls how much the teacher intervenes, while uncertainty determines where it intervenes. We recompute the threshold at the start of each

episode and hold it fixed within the episode. This design connects to uncertainty-guided expert intervention (Menda et al., 2019) and budget-aware quantile thresholding (Hoque et al., 2022) in interactive imitation learning.

At turn t, the student samples a proposed action $a_t^S \sim \pi_\theta(\cdot \mid h_t)$ and UOPD computes its uncertainty score $\delta_t = \mathcal{U}(\pi_T, \pi_\theta; h_t, a_t^S)$. If $\delta_t > \tau_n$, UOPD samples a teacher action $a_t^T \sim \pi_T(\cdot \mid h_t)$ and executes it in place of a_t^S; otherwise, it executes the student action. The executed action a_t is appended to the history along with the resulting observation.

We gradually decrease the target intervention rate to provide more teacher guidance early in training and progressively shift rollout control to the student:

$$r_n = r_{\text{start}} + (r_{\text{end}} - r_{\text{start}}) \min\left(\frac{n}{N_{\text{decay}}}, 1\right), \tag{2}$$

where r_{start} and r_{end} denote the initial and final target rates, and N_{decay} denotes the decay duration.

Intervention-Conditioned Distillation. At a non-intervened turn, the student receives the standard OPD supervision on a_t^S. At an intervened turn, the executed action a_t^T is sampled from the teacher. The standard OPD term relies on student-policy samples and therefore cannot be applied directly to a_t^T: at a fixed history h_t, its expectation under teacher sampling becomes

$$\mathbb{E}_{a_t^T \sim \pi_T(\cdot|h_t)}\left[\log \frac{\pi_\theta(a_t^T \mid h_t)}{\pi_T(a_t^T \mid h_t)}\right] = -\mathrm{KL}(\pi_T(\cdot \mid h_t) \,\|\, \pi_\theta(\cdot \mid h_t)). \tag{3}$$

We instead minimize the *forward* KL from teacher to student at intervened turns through SFT, since

$$\mathbb{E}_{a_t^T \sim \pi_T(\cdot|h_t)}[-\log \pi_\theta(a_t^T \mid h_t)] = \mathcal{H}(\pi_T(\cdot \mid h_t)) + \mathrm{KL}(\pi_T(\cdot \mid h_t) \,\|\, \pi_\theta(\cdot \mid h_t)).$$

Here $\mathcal{H}(p) := -\mathbb{E}_{a\sim p}[\log p(a)]$ denotes the entropy of an action distribution p. The teacher entropy is independent of θ, so SFT and forward KL have the same expected gradient. Combining OPD on student actions with SFT on teacher actions yields

$$\ell_t^{\text{UOPD}} = \begin{cases} \ell_t^{\text{OPD}}(a_t^S; \theta), & \delta_t \le \tau_n, \\ -\beta \log \pi_\theta(a_t^T \mid h_t), & \delta_t > \tau_n, \end{cases} \tag{4}$$

where β controls the weight of the teacher-action SFT loss. The intervention decision thus determines both the executed action and the KL direction at each turn.

Appendix B illustrates the mode-seeking and mode-covering tendencies of reverse and forward KL, respectively.

4.3 Understanding Intervention and Imitation

The intervention mechanism in Section 4.2 gives each teacher correction two roles: an action to execute and a target to imitate. We first characterize how executing selected corrections changes rollout return. We then show how teacher-action imitation can improve the student's decision at a step where reverse-KL supervision would reduce return.

4.3.1 Improving Rollout Quality

We measure rollout quality by the expected episode return $J(\pi)$ and write $\pi_S := \pi_\theta$ for the current student. During collection, fix π_S and the intervention threshold τ_n. The student proposes $a_t^S \sim \pi_S(\cdot \mid h_t)$, and $I_t = \mathbf{1}\{\mathcal{U}(\pi_T, \pi_S; h_t, a_t^S) > \tau_n\}$ determines whether a teacher action $a_t^T \sim \pi_T(\cdot \mid h_t)$ replaces it. Let π_G denote this assisted execution policy and $N_{\text{int}} := \sum_t I_t$. The student and assisted policies interact with the same environment and initial task distribution. Write $Q_t^{\pi_S}(h, a)$ for the expected remaining return from taking a at (t, h) and then following the student. Expectations under π_G include the student proposals as well as the executed actions.

Assumption 1 (Average recoverability)**.** *Whenever $\mathbb{E}_{\pi_G}[N_{\text{int}}] > 0$, teacher actions have average advantage of at least $\gamma > 0$ at the selected turns:*

$$\frac{\mathbb{E}_{\pi_G}\left[\sum_t I_t\big(Q_t^{\pi_S}(h_t, a_t^T) - Q_t^{\pi_S}(h_t, a_t^S)\big)\right]}{\mathbb{E}_{\pi_G}[N_{\text{int}}]} \ge \gamma. \tag{5}$$

Algorithm 1 UOPD

Input: student π_θ, frozen teacher π_T, environment E, uncertainty score $\mathcal{U}$, uncertainty score buffer $\mathcal{D}_0$, rate parameters $(r_{\text{start}}, r_{\text{end}}, N_{\text{decay}})$, SFT weight β.

for each training iteration $n = 0, 1, \ldots$ **do**
  Compute target intervention rate r_n using Eq. 2.
  for each episode in the rollout batch, in parallel **do**
    Set $\tau_n \leftarrow \text{Quantile}_{1-r_n}(\mathcal{D}_n)$; hold it fixed for this episode.
    Reset E for task input q; set $h_0 \leftarrow q$ and $t \leftarrow 0$.
    while the episode is not terminated **do**
      Sample $a_t^S \sim \pi_\theta(\cdot \mid h_t)$; compute and record $\delta_t = \mathcal{U}(\pi_T, \pi_\theta; h_t, a_t^S)$.
      If $\delta_t > \tau_n$, sample $a_t^T \sim \pi_T(\cdot \mid h_t)$ and set $a_t \leftarrow a_t^T$; otherwise set $a_t \leftarrow a_t^S$.
      Compute ℓ_t^{UOPD} using Eq. 4.
      Execute a_t, observe o_{t+1}, and set $h_{t+1} \leftarrow (h_t, a_t, o_{t+1})$; $t \leftarrow t + 1$.
    end while
    Update $\mathcal{D}_n$ with this episode's recorded uncertainty scores.
  end for
  Update the student π_θ using the collected batch; carry the updated buffer to $\mathcal{D}_{n+1}$.
end for

This condition averages over the selected turns and the histories reached by π_G; it does not require every teacher correction to improve return.

Theorem 1 (Return improvement under selective intervention)**.** *For any uncertainty intervention rule and teacher policy in a finite-horizon environment,*

$$J(\pi_G) - J(\pi_S) = \mathbb{E}_{\pi_G}\left[\sum_t I_t\big(Q_t^{\pi_S}(h_t, a_t^T) - Q_t^{\pi_S}(h_t, a_t^S)\big)\right]. \tag{6}$$

Under Assumption 1, this identity gives

$$J(\pi_G) - J(\pi_S) \geq \gamma\, \mathbb{E}_{\pi_G}[N_{\text{int}}] \geq 0. \tag{7}$$

Theorem 1 relates rollout improvement to the downstream value of the selected corrections. Each term compares a teacher action with the student proposal it replaces, while the expectation accounts for the histories reached after earlier interventions. Under average recoverability, the gain is at least γ per intervention in expectation. This is the execution benefit illustrated by Fig. 2(c): an action correction improves the subsequent rollout before any student update. The proof is in Appendix A.1.

4.3.2 Learning from Teacher Corrections

Beyond changing the rollout, intervention supplies a teacher action as the learning target at the current turn. We examine how this changes the student's update at a decision step. Reverse KL can penalize an overproduced action without directing the resulting probability change toward higher-return actions. We show that teacher-action SFT can improve the update direction even when the student can represent the teacher exactly. At a fixed turn t and history h, let $J_{t,h}(\pi) := \mathbb{E}_{a\sim\pi(\cdot|h)}[Q_t^{\pi_S}(h, a)]$ measure the return of an action distribution, keeping π_S as the continuation policy.

Proposition 1 (Benefit of teacher-action imitation)**.** *There exist a one-step, three-action decision problem with binary terminal rewards, full-support student and teacher policies with $J_{t,h}(\pi_T) > J_{t,h}(\pi_S)$, and an unrestricted softmax student $\pi_\theta(\cdot \mid h) = \text{softmax}(\theta)$ initialized at π_S. Let π^+_{OPD} and π^+_{SFT} result from one ordinary gradient step on the action logits, starting from the same student and using the same step size α, with the population gradients of reverse KL and teacher-action SFT at h, respectively. For all sufficiently small $\alpha > 0$,*

$$J_{t,h}(\pi^+_{\text{OPD}}) < J_{t,h}(\pi_S) < J_{t,h}(\pi^+_{\text{SFT}}). \tag{8}$$

Proposition 1 identifies a decision step where the supervision objective determines whether an update improves return. In the construction, reverse KL reduces an overproduced successful action but transfers some of its probability to an unsuccessful action. Teacher-action SFT instead reduces the unsuccessful action's probability. This motivates a teacher imitation target at intervention turns within UOPD, which retains OPD supervision on student-executed actions. The parameterized construction and full proof are in Appendix A.2.

Table 1: Main results on ALFWorld and WebShop. We report success rate (SR, %) on ALFWorld, and mean task score, SR (%) on WebShop. We also include average number of turns. Results are means ± standard deviations over three evaluation seeds. Bold indicates the best result among distillation methods.

	ALFWorld				WebShop		
	Seen		Unseen		Test		
Method	SR ↑	Turns ↓	SR ↑	Turns ↓	Score ↑	SR ↑	Turns ↓
RL-Qwen2.5-7B teacher → Qwen2.5-3B student							
Student (zero-shot)	21.0 ±0.8	43.8 ±0.3	15.9 ±3.4	46.0 ±1.1	12.2 ±2.1	1.0 ±0.5	13.6 ±0.3
Teacher (zero-shot)	97.4 ±0.4	9.1 ±0.1	92.8 ±0.4	12.3 ±0.4	85.4 ±0.6	77.9 ±1.2	6.6 ±0.3
OPD	88.1 ±1.8	14.2 ±0.3	85.3 ±1.6	16.3 ±0.6	77.5 ±2.2	66.9 ±3.0	7.4 ±0.3
TCOD-F2B	85.7 ±1.2	14.2 ±0.2	87.1 ±0.9	15.6 ±0.7	76.4 ±1.8	64.3 ±1.2	7.4 ±0.2
TCOD-B2F	86.4 ±2.1	14.1 ±0.6	85.6 ±1.1	15.9 ±0.4	76.3 ±0.5	65.1 ±2.4	7.5 ±0.1
FTB-OPD	87.4 ±1.1	14.1 ±0.3	86.6 ±2.2	**14.7** ±0.7	77.3 ±0.3	69.0 ±1.6	7.4 ±0.1
UOPD	**90.0** ±0.7	**13.6** ±0.3	**88.8** ±1.3	14.9 ±0.8	**82.4** ±2.0	**69.3** ±2.0	**6.5** ±0.1
RL-Qwen2.5-7B teacher → Qwen2.5-1.5B student							
Student (zero-shot)	6.2 ±0.4	48.5 ±0.3	3.7 ±0.7	49.2 ±0.2	22.6 ±3.8	2.3 ±2.1	10.9 ±0.4
Teacher (zero-shot)	97.4 ±0.4	9.1 ±0.1	92.8 ±0.4	12.3 ±0.4	85.4 ±0.6	77.9 ±1.2	6.6 ±0.3
OPD	86.2 ±2.9	15.2 ±1.0	84.8 ±0.4	17.2 ±0.9	66.3 ±0.9	52.3 ±1.4	7.9 ±0.4
TCOD-F2B	86.0 ±1.8	14.3 ±0.8	85.1 ±2.0	16.8 ±1.0	69.5 ±1.2	51.8 ±2.5	7.2 ±0.2
TCOD-B2F	84.5 ±0.4	15.2 ±0.1	84.1 ±0.9	18.0 ±0.1	68.4 ±1.2	54.2 ±0.5	7.6 ±0.1
FTB-OPD	85.2 ±1.1	15.5 ±0.4	83.6 ±1.3	17.7 ±0.4	67.7 ±0.5	55.2 ±1.2	9.4 ±0.1
UOPD	**89.8** ±0.8	**13.0** ±0.5	**86.6** ±0.7	**15.3** ±0.2	**76.8** ±1.8	**58.6** ±0.0	**6.8** ±0.4

5 Experiments

5.1 Experimental Setup

Benchmarks and metrics. We evaluate on three multi-turn agent benchmarks: ALFWorld (Shridhar et al., 2021), WebShop (Yao et al., 2022), and Search (Jin et al., 2025). These benchmarks cover embodied household tasks, web shopping, and search-based question answering, allowing us to assess the effectiveness and efficiency of our uncertainty-guided intervention across different environments. We report success rate (SR) on both the seen and unseen splits of ALFWorld, task score and success rate on WebShop, and exact match on four multi-hop Search datasets. Average interaction turns measure efficiency. Dataset and evaluation details are provided in Appendix D.

Baselines. We compare UOPD with four baselines. **OPD** (Agarwal et al., 2024; Lu & Thinking Machines Lab, 2025) applies reverse-KL supervision to student-generated actions throughout a rollout. **TCOD-F2B** (WANG et al., 2026) progressively extends the student's rollout horizon from the initial state. **TCOD-B2F** (WANG et al., 2026) starts student rollouts after expert action prefixes and gradually shortens these prefixes to expose the student to earlier decisions. **FTB-OPD** (Chen et al., 2026) introduces teacher corrections at high-disagreement steps and retains them for distillation when they improve teacher preference over subsequent student continuations.

Training details. On ALFWorld and WebShop, we use Qwen2.5-3B-Instruct and Qwen2.5-1.5B-Instruct (Yang et al., 2024) as students, with task-specific GiGPO-Qwen2.5-7B-Instruct teachers trained using reinforcement learning (Feng et al., 2025). For Search, we distil a base Qwen2.5-1.5B student from SearchR1-Qwen2.5-7B-em-ppo (Jin et al., 2025). For all methods, we train for 250 steps on ALFWorld and 150 on WebShop with rollout batch size 16, and for 300 steps on Search with batch size 64. For ALFWorld and WebShop, UOPD uses teacher confidence as its uncertainty signal. More training configurations and implementation details are provided in Appendix D.

5.2 Main Results

We evaluate UOPD's task performance and efficiency on ALFWorld and WebShop (Table 1) and multi-hop question answering (Fig. 3).

UOPD improves task performance and interaction efficiency. Across both student sizes, UOPD achieves the highest success rates on the ALFWorld seen and unseen splits and the highest WebShop scores among the compared distillation methods. The gains on unseen ALFWorld tasks show that the benefit extends to environments beyond those encountered during training, while the improvements at 1.5B demonstrate effective transfer across a larger teacher–student capacity gap. On WebShop, UOPD raises the 1.5B student's score from OPD's 66.3 to 76.8 and success rate from 52.3% to 58.6%, while reducing average turns from 7.9 to 6.8. UOPD requires the fewest interaction turns in five of the six evaluation settings, demonstrating its efficiency across tasks and student sizes.

The gains extend to multi-hop search. Multi-hop search requires the model to retrieve relevant evidence and combine it across multiple reasoning steps. With a 1.5B student, UOPD consistently outperforms OPD on all four datasets, improving performance by 2.67–8.00 percentage points, demonstrating the benefit of uncertainty-guided supervision for search tasks. Meanwhile, the average number of turns decreases from 5.19 to 3.90, indicating that improved accuracy is accompanied by higher efficiency at inference time.

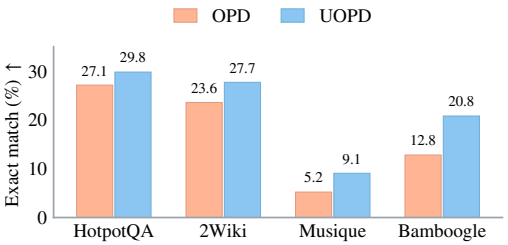


Figure 3: Multi-hop Search results.

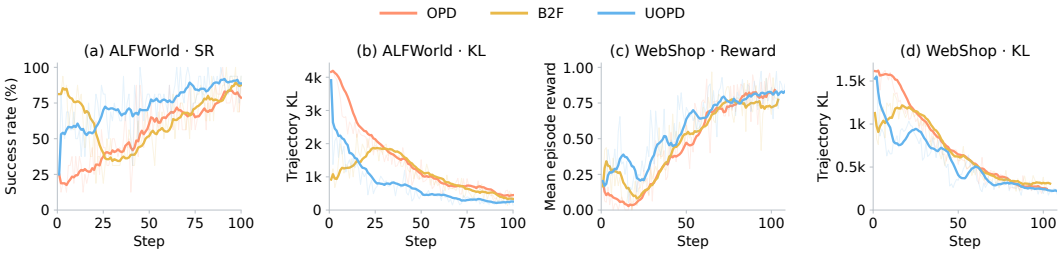


Figure 4: Training dynamics with 3B students. We compare OPD, TCOD-B2F (B2F), and UOPD on ALFWorld (a,b: success rate and teacher–student KL) and WebShop (c,d: reward and teacher–student KL).

Effective guidance during training. Fig. 4 shows that UOPD improves rollout quality early in training, achieving higher success rates on ALFWorld and higher rewards on WebShop than OPD. UOPD also exhibits a faster decline in trajectory-level teacher–student KL than OPD and B2F. B2F improves initial rollout quality through teacher-executed prefixes, but its performance drops as these prefixes shorten. This suggests that learning from teacher-provided states may not prepare the student to execute longer action sequences independently. UOPD targets high-uncertainty turns throughout the rollout, providing teacher corrections while letting the student act and learn throughout the task.

5.3 Additional Analysis

Table 2 presents ablation studies with 3B students on ALFWorld and WebShop, examining how individual components of UOPD contribute to task performance and interaction efficiency.

Selecting intervention turns. We replace uncertainty-based selection with random intervention, sampling each turn with the same target-rate schedule while retaining teacher execution and the SFT loss ($\beta = 1$). Compared with random intervention, UOPD improves unseen ALFWorld success from 83.8% to 88.8% and WebShop score from 77.3 to 82.4, while reducing WebShop turns from 7.4 to 6.5. Using the length-normalized student–teacher log-probability gap as the intervention signal also yields higher success rates than random selection on both environments. Teacher confidence achieves comparable success rates while providing higher task scores and shorter interactions on WebShop.

Learning from teacher corrections. Removing the SFT loss reduces performance on both benchmarks, with the largest success-rate drop of 8.0 percentage points on unseen ALFWorld. Teacher execution alone therefore does not recover the full benefit of learning from corrections. The default $\beta = 1$ performs best among the tested weights; increasing it to $\beta = 2$ brings no further gains.

Executing teacher corrections. On WebShop, teacher execution increases the score from 77.6 to 82.4 and reduces average turns from 7.2 to 6.5, while success rates remain similar. The benefit

Table 2: Effects of intervention selection, SFT weight, teacher execution, and schedule with a 3B student. The default UOPD configuration uses teacher confidence, $\beta = 1$, teacher execution, and a $30\% \rightarrow 5\%$ intervention schedule. Random intervention samples each turn with the scheduled probability. Values are means over three evaluation seeds, and bold marks the best value in each column.

	ALFWorld		WebShop		
Configuration	Seen SR ↑	Unseen SR ↑	Score ↑	SR ↑	Turns ↓
UOPD ($\beta = 1$, $30\% \rightarrow 5\%$)	90.0	88.8	**82.4**	**69.3**	**6.5**
Intervention signal					
Confidence gap	89.8	87.8	77.3	69.0	7.2
Random intervention	88.3	83.8	77.3	64.6	7.4
SFT weight					
$\beta = 0$	86.2	80.8	79.7	66.1	6.7
$\beta = 0.5$	85.7	86.3	78.0	68.0	7.3
$\beta = 2$	89.5	86.8	77.3	68.0	7.5
Teacher execution					
w/o teacher execution	89.5	84.3	77.6	69.0	7.2
Intervention schedule					
Constant 15%	90.0	**89.6**	78.0	67.7	7.0
$50\% \rightarrow 30\%$	**90.5**	86.6	69.9	59.4	8.2

thus extends beyond whether a task is fully completed: the student better satisfies the shopping requirements and uses fewer steps per episode. This suggests that executing corrections during training can complement the learning signal provided by the SFT loss.

Allocating teacher intervention. A higher intervention rate does not consistently improve performance. The $50\% \rightarrow 30\%$ schedule slightly improves seen ALFWorld success but lowers unseen success and performs worse on WebShop. A constant 15% rate remains competitive on ALFWorld, achieving the highest unseen success among the tested schedules, whereas $30\% \rightarrow 5\%$ yields the highest score and fewest turns on WebShop. See Appendix F for intervention dynamics.

Training efficiency. Fig. 5 compares training time at equal step counts per benchmark. UOPD reduces training time relative to FTB-OPD by 13.8% on ALFWorld and 36.5% on WebShop, with 11.5% and 14.7% overhead over OPD. UOPD selects corrections before action execution, avoiding the paired student continuations used by FTB-OPD to validate teacher guidance. It thus improves performance with modest training cost compared with standard on-policy distillation.

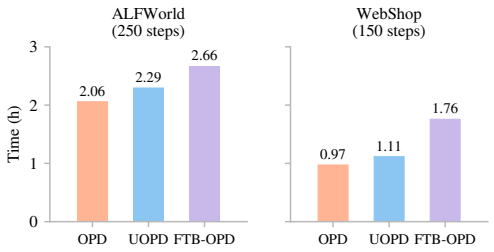


Figure 5: Training time with 3B students.

6 Conclusion

In this work, we investigate when teacher intervention improves the trajectories used for on-policy distillation of multi-turn agents. Our controlled experiments show that a teacher correction at a low-confidence step improves subsequent student behavior and task success, outperforming random intervention. Building on this finding, we propose UOPD, which uses teacher uncertainty to select turns for action correction and imitation learning while retaining standard OPD on student actions. Experiments on ALFWorld, WebShop, and multi-hop Search demonstrate that UOPD improves task performance over OPD while reducing interaction turns.

Our findings highlight the value of teacher uncertainty for guiding both what an agent learns and the trajectories from which it learns. Selective corrections provide useful learning targets and steer subsequent interaction toward better outcomes. Future work could extend this approach to longer-horizon tasks and adapt the intervention budget to task difficulty and student progress.

AI USE STATEMENT

We used AI assistants to support code development, manuscript drafting and editing, and figure preparation. The authors reviewed and verified the resulting code, analyses, and text, and take full responsibility for the paper's accuracy, originality, and conclusions.

REPRODUCIBILITY STATEMENT

Section 4 details our method, with the overall procedure summarized in Algorithm 1. Appendix C details the controlled ALFWorld study. Appendix D documents datasets, training configurations, baseline implementations, evaluation protocols, and computational resources. Appendix E provides environment prompts and response formats.

REFERENCES

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

Rishabh Agrawal, Jacob Fein-Ashley, and Paria Rashidinejad. Reinforcement learning from rich feedback with distributional DAgger. *arXiv preprint arXiv:2606.05152*, 2026.

Chishui Chen, Yaoyou Fan, Te Sun, Yi Yang, Chenghao Sun, Delin Mao, Hongbo Qiao, Zuowei Zhang, Junxi Wang, Chenxing Sun, Yangen Hu, Lu Pan, Xuyang Liu, and Linfeng Zhang. Look ahead before you distill: Future trajectory validation of teacher guidance for agentic on-policy distillation. *arXiv preprint arXiv:2608.01953*, 2026.

Junfeng Fang, Zhepei Hong, Mao Zheng, Mingyang Song, Gengsheng Li, Houcheng Jiang, Dan Zhang, Haiyun Guo, Xiang Wang, and Tat-Seng Chua. Rubric-based on-policy distillation. *arXiv preprint arXiv:2605.07396*, 2026.

Lang Feng, Zhenghai Xue, Tingcong Liu, and Bo An. Group-in-group policy optimization for LLM agent training. *arXiv preprint arXiv:2505.10978*, 2025.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Ryan Hoque, Ashwin Balakrishna, Ellen Novoseller, Albert Wilcox, Daniel S Brown, and Ken Goldberg. Thriftydagger: Budget-aware novelty and risk gating for interactive imitation learning. In *Conference on Robot Learning*, pp. 598–608. PMLR, 2022.

Nan Jia, Haojin Yang, Xing Ma, Jiesong Lian, Shuailiang Zhang, Weipeng Zhang, Ke Zeng, Xunliang Cai, and Zequn Sun. Asymmetric on-policy distillation: Bridging exploitation and imitation at the token level. *arXiv preprint arXiv:2605.06387*, 2026.

Li Jiang, Haoran Xu, Yichuan Ding, and Amy Zhang. Trajectory-refined distillation. *arXiv preprint arXiv:2606.08432*, 2026.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Arik, Dong Wang, Hamed Zamani, and Jiawei Han. Search-r1: Training LLMs to reason and leverage search engines with reinforcement learning. *arXiv preprint arXiv:2503.09516*, 2025.

Woogyeol Jin, Taywon Min, Yongjin Yang, Dennis Wei, Yi Zhou, Swanand Ravindra Kadhe, Nathalie Baracaldo, and Kimin Lee. Entropy-aware on-policy distillation of language models. *arXiv preprint arXiv:2603.07079*, 2026.

Junlong Ke, Zichen Wen, Weijia Li, Conghui He, and Linfeng Zhang. Respecting self-uncertainty in on-policy self-distillation for efficient LLM reasoning. *arXiv preprint arXiv:2605.13255*, 2026.

Michael Kelly, Chelsea Sidrane, Katherine Driggs-Campbell, and Mykel J. Kochenderfer. Hg-dagger: Interactive imitation learning with human experts. In *International Conference on Robotics and Automation (ICRA)*, 2019.

Gengsheng Li, Mao Zheng, Mingyang Song, Ruiqi Liu, Tianyu Yang, Jie Sun, Qiyong Zhong, Haiyun Guo, Junfeng Fang, Dan Zhang, and Jinqiao Wang. On-policy distillation with curriculum turn-level guidance for multi-turn agents. *arXiv preprint arXiv:2606.15912*, 2026a.

Xiaozhe Li, Tianyi Lyu, Yang Li, Yichuan Ma, Peiji Li, Linyang Li, Qipeng Guo, Dahua Lin, and Kai Chen. What and when to distill: Selective hindsight distillation for multi-turn agents. *arXiv preprint arXiv:2605.19447*, 2026b.

Yaxuan Li, Yuxin Zuo, Bingxiang He, Jinqian Zhang, Chaojun Xiao, Cheng Qian, Tianyu Yu, Huanang Gao, Wenkai Yang, Zhiyuan Liu, and Ning Ding. Rethinking on-policy distillation of large language models: Phenomenology, mechanism, and recipe. *arXiv preprint arXiv:2604.13016*, 2026c.

Kun Liang, Chenming Tang, Clive Bai, Weijie Liu, Saiyong Yang, and Yunfang Wu. ADWIN: Adaptive windows for horizon-aware on-policy distillation. *arXiv preprint arXiv:2605.28396*, 2026.

Baohao Liao, Hanze Dong, Christof Monz, Xinxing Xu, Li Dong, and Furu Wei. Multi-turn on-policy distillation with prefix replay. *arXiv preprint arXiv:2607.04763*, 2026.

Yanjiang Liu, Jie Lou, Xinyan Guan, Yuqiu Ji, Hongyu Lin, Ben He, Xianpei Han, Le Sun, Xing Yu, and Yaojie Lu. Your teacher can't help you here: Combating supervision fidelity decay in on-policy distillation. *arXiv preprint arXiv:2605.30833*, 2026.

Kevin Lu and Thinking Machines Lab. On-policy distillation. *Thinking Machines Lab: Connectionism*, 2025. doi: 10.64434/tml.20251026. URL `https://thinkingmachines.ai/blog/on-policy-distillation/`.

Zhengxi Lu, Zhiyuan Yao, Zhuowen Han, Zi-Han Wang, Jinyang Wu, Qi Gu, Xunliang Cai, Weiming Lu, Jun Xiao, Yueting Zhuang, and Yongliang Shen. Self-distilled agentic reinforcement learning. *arXiv preprint arXiv:2605.15155*, 2026.

Kunal Menda, Katherine Driggs-Campbell, and Mykel J Kochenderfer. Ensembledagger: A bayesian approach to safe imitation learning. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5041–5048. IEEE, 2019.

Minjae Oh, Sangjun Song, Gyubin Choi, Yunho Choi, and Yohan Jo. KL for a KL: On-policy distillation with control variate baseline. *arXiv preprint arXiv:2605.07865*, 2026.

Xuchen Pan, Yanxi Chen, Yushuo Chen, Yuchang Sun, Daoyuan Chen, Wenhao Zhang, Yuexiang Xie, Yilun Huang, Yilei Zhang, Dawei Gao, Yaliang Li, Bolin Ding, and Jingren Zhou. Trinity-RFT: A general-purpose and unified framework for reinforcement fine-tuning of large language models. *arXiv preprint arXiv:2505.17826*, 2025. URL `https://arxiv.org/abs/2505.17826`.

Zikun Qu, Min Zhang, Mingze Kong, Zhiwei Shang, Zhengyu Chen, Yikun Ban, Shuang Qiu, and Zhongxiang Dai. SPOT: Sparse probing and outcome calibration for on-policy distillation. *arXiv preprint arXiv:2608.04419*, 2026.

Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2011.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. ALFWorld: Aligning text and embodied environments for interactive learning. In *International Conference on Learning Representations (ICLR)*, 2021.

Aasheesh Singh, Vishal Vaddina, and Dagnachew Birru. ORPO-distill: Mixed-policy preference optimization for cross-architecture LLM distillation. *arXiv preprint arXiv:2509.25100*, 2025.

Giacomo Spigler. Proximal policy distillation. *arXiv preprint arXiv:2407.15134*, 2025.

Bing Wang, Shaotian Yan, Chen Shen, Kaiyuan Liu, Sinan Fan, Ximing Li, Rui Miao, Xiaosong Yuan, Zhanming Shen, and Jieping Ye. Backtracking when it strays: Mitigating dual exposure biases in LLM reasoning distillation. *arXiv preprint arXiv:2605.19433*, 2026.

Jiaqi WANG, Wenhao Zhang, Weijie Shi, Yaliang Li, and James Cheng. Exploring temporal curriculum in on-policy distillation for multi-turn autonomous agents. In *Third Conference on Language Modeling*, 2026. URL https://openreview.net/forum?id=KH9eJRz8WR.

Yuanyi Wang, Su Lu, Yanggan Gu, Pengkai Wang, Yifan Yang, Zhaoyi Yan, Congkai Xie, Jianmin Wu, and Hongxia Yang. Not all disagreement is learnable: Token teachability in on-policy distillation. *arXiv preprint arXiv:2605.26844*, 2026.

Hongling Xu, Qi Zhu, Heyuan Deng, Jinpeng Li, Lu Hou, Yasheng Wang, Lifeng Shang, Ruifeng Xu, and Fei Mi. KDRL: Post-training reasoning LLMs via unified knowledge distillation and reinforcement learning. *arXiv preprint arXiv:2506.02208*, 2025a.

Wenda Xu, Rujun Han, Zifeng Wang, Long T. Le, Dhruv Madeka, Lei Li, William Yang Wang, Rishabh Agarwal, Chen-Yu Lee, and Tomas Pfister. Speculative knowledge distillation: Bridging the teacher-student gap through interleaved sampling. In *International Conference on Learning Representations (ICLR)*, 2025b.

Yuanda Xu, Hejian Sang, Zhengze Zhou, Ran He, Zhipeng Wang, and Alborz Geramifard. Beyond GRPO and on-policy distillation: An empirical sparse-to-dense reward principle for language-model post-training. *arXiv preprint arXiv:2605.12483*, 2026.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, et al. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024. URL https://arxiv.org/abs/2412.15115.

Wenkai Yang, Weijie Liu, Ruobing Xie, Kai Yang, Saiyong Yang, and Yankai Lin. Learning beyond teacher: Generalized on-policy distillation with reward extrapolation. *arXiv preprint arXiv:2602.12125*, 2026a.

Zhicheng Yang, Zhijiang Guo, Yifan Song, Minrui Xu, Yongxin Wang, Yiwei Wang, Xiaodan Liang, and Jing Tang. Prune-OPD: Efficient and reliable on-policy distillation for long-horizon reasoning. *arXiv preprint arXiv:2605.07804*, 2026b.

Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. WebShop: Towards scalable real-world web interaction with grounded language agents. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Weichen Yu, Xiaomin Li, Yizhou Zhao, Xiaoze Liu, Ruowang Zhang, Haixin Wang, Yinyi Luo, Chen Henry Wu, Gaurav Mittal, Matt Fredrikson, and Yu Hu. Multi-rollout on-policy distillation via peer successes and failures. *arXiv preprint arXiv:2605.12652*, 2026.

Ke Zhang, Yunjie Tian, Dongdi Zhao, Yijiang Li, Yuanye Liu, Vishal M. Patel, and Di Fu. On-policy distillation with best-of-N teacher rollout selection. *arXiv preprint arXiv:2605.09725*, 2026a.

Wenhao Zhang, Yuexiang Xie, Yuchang Sun, Yanxi Chen, Guoyin Wang, Yaliang Li, Bolin Ding, and Jingren Zhou. On-policy RL meets off-policy experts: Harmonizing supervised fine-tuning and reinforcement learning via dynamic weighting. *arXiv preprint arXiv:2508.11408*, 2026b.

Yaocheng Zhang, Jiajun Chai, Yuqian Fu, Songjun Tu, Xiaohan Wang, Wei Lin, Guojun Yin, Qichao Zhang, Yuanheng Zhu, and Dongbin Zhao. Are full rollouts necessary for on-policy distillation? *arXiv preprint arXiv:2605.31490*, 2026c.

Zhaoyang Zhang, Shuli Jiang, Yantao Shen, Yuting Zhang, Dhananjay Ram, Shuo Yang, Zhuowen Tu, Wei Xia, and Stefano Soatto. Reinforcement-aware knowledge distillation for LLM reasoning. *arXiv preprint arXiv:2602.22495*, 2026d.

Anhao Zhao, Haoran Xin, Yingqi Fan, Junlong Tong, Wenjie Li, and Xiaoyu Shen. Decoupling KL and trajectories: A unified perspective for SFT, DAgger, offline RL, and OPD in LLM distillation. *arXiv preprint arXiv:2605.16826*, 2026.

Qiyong Zhong, Mao Zheng, Mingyang Song, Xin Lin, Jie Sun, Houcheng Jiang, Xiang Wang, and Junfeng Fang. SOD: Step-wise on-policy distillation for small language model agents. *arXiv preprint arXiv:2605.07725*, 2026.

Yuhang Zhou, Lizhu Zhang, Yifan Wu, Mingyi Wang, Bo Peng, Jiayi Liu, Xiangjun Fan, and Zhuokai Zhao. SAGE-OPD: Selective agent-guided intervention for multi-turn on-policy distillation. *arXiv preprint arXiv:2606.19659*, 2026a.

Yuhang Zhou, Kai Zheng, Haoling Li, Dengyun Peng, Can Xu, and Jingjing Chen. TurnOPD: Making on-policy distillation turn-aware for efficient long-horizon agent training. *arXiv preprint arXiv:2607.05804*, 2026b.

Siqi Zhu, Xuyan Ye, Hongyu Lu, Weiye Shi, and Ge Liu. The many faces of on-policy distillation: Pitfalls, mechanisms, and fixes. *arXiv preprint arXiv:2605.11182*, 2026a.

Wenhong Zhu, Ruobing Xie, Rui Wang, and Pengfei Liu. Hybrid policy distillation for LLMs. *arXiv preprint arXiv:2604.20244*, 2026b.

A PROOFS

A.1 PROOF OF THEOREM 1

Proof. Define the student's finite-horizon value and advantage functions by

$$V_t^{\pi_S}(h) := \mathbb{E}_{a\sim\pi_S(\cdot|h)} Q_t^{\pi_S}(h,a),$$
$$A_t^{\pi_S}(h,a) := Q_t^{\pi_S}(h,a) - V_t^{\pi_S}(h),$$

with $V_H^{\pi_S} = 0$.

Step 1: derive the finite-horizon performance-difference identity. By the Bellman definition of $Q_t^{\pi_S}$,

$$Q_t^{\pi_S}(h_t,a_t) = \mathbb{E}\big[r_t + V_{t+1}^{\pi_S}(h_{t+1}) \mid h_t, a_t\big].$$

Taking expectation under the assisted trajectory and summing over turns gives

$$\begin{aligned}\sum_{t=0}^{H-1} \mathbb{E}_{\pi_G}[A_t^{\pi_S}(h_t,a_t)] &= \sum_{t=0}^{H-1} \mathbb{E}_{\pi_G}\big[r_t + V_{t+1}^{\pi_S}(h_{t+1}) - V_t^{\pi_S}(h_t)\big] \\ &= J(\pi_G) - J(\pi_S).\end{aligned}$$

The value terms telescope because the policies share the distribution of initial task inputs and $V_H^{\pi_S} = 0$.

Step 2: evaluate the advantage at an intervention turn. Condition on (t,h_t) and sample the student proposal and teacher action as in the proposal-coupled construction. Since $a_t = a_t^S$ when $I_t = 0$ and $a_t = a_t^T$ when $I_t = 1$,

$$\begin{aligned}&\mathbb{E}[A_t^{\pi_S}(h_t,a_t) \mid h_t] \\ &= \mathbb{E}\big[I_t A_t^{\pi_S}(h_t,a_t^T) + (1-I_t)A_t^{\pi_S}(h_t,a_t^S) \,\big|\, h_t\big] \\ &= \mathbb{E}\big[I_t\big(A_t^{\pi_S}(h_t,a_t^T) - A_t^{\pi_S}(h_t,a_t^S)\big) \,\big|\, h_t\big] \\ &= \mathbb{E}\big[I_t\big(Q_t^{\pi_S}(h_t,a_t^T) - Q_t^{\pi_S}(h_t,a_t^S)\big) \,\big|\, h_t\big].\end{aligned}$$

The third line uses $\mathbb{E}[A_t^{\pi_S}(h_t,a_t^S) \mid h_t] = 0$; the final line cancels the common value term.

Step 3: aggregate over the assisted trajectory. Substituting Step 2 into Step 1 and applying the law of total expectation gives Eq. 6.

Under Assumption 1, multiplying Eq. 5 by $\mathbb{E}_{\pi_G}[N_{\text{int}}]$ and applying the theorem yields

$$J(\pi_G) - J(\pi_S) \geq \gamma\, \mathbb{E}_{\pi_G}[N_{\text{int}}] \geq 0,$$

which proves the lower bound in Theorem 1. If the expected number of interventions is zero, $\pi_G = \pi_S$ almost surely and the same conclusion holds directly. □

A.2 PROOF OF PROPOSITION 1

Proof. Fix a history h at a terminal decision turn t. There are three actions $\{a,b,c\}$ with rewards $r(a) = r(b) = 1$ and $r(c) = 0$, so $Q_t^{\pi_S}(h,i) = r(i)$. Suppress h in the distributions and use the unrestricted softmax parameterization

$$p_i(\theta) = \frac{\exp(\theta_i)}{\sum_{j\in\{a,b,c\}} \exp(\theta_j)}.$$

Initialize $\theta_0 = (0,0,0)$, so $p = \pi_S = (1/3,1/3,1/3)$. The three logits are independent parameters. Choose a teacher

$$u = \pi_T = \big(v(1-\epsilon), v\epsilon, 1-v\big), \qquad \frac{2}{3} < v < 1, \quad 0 < \epsilon < \frac{1}{2},$$

with

$$v^2\epsilon(1-\epsilon) < (1-v)^2. \tag{9}$$

For any $v \in (2/3, 1)$, this condition holds for sufficiently small positive ϵ. Thus the construction describes a family of full-support teachers, each with return $v > 2/3 = J_{t,h}(\pi_S)$. Every teacher in this family is represented by the student class, for example by logits $\theta_i = \log u_i$.

Write $\ell_i = \log(p_i/u_i)$ and $\bar{\ell} = \sum_i p_i \ell_i$. The ordinary logit gradients of the two losses in Proposition 1 are

$$G_{\mathrm{OPD},i} = p_i(\ell_i - \bar{\ell}), \qquad G_{\mathrm{SFT},i} = p_i - u_i. \tag{10}$$

Both gradients sum to zero. The first follows by differentiating $\sum_i p_i \log(p_i/u_i)$ and using $\nabla_\theta \log p_i = e_i - p$; the second is the expected teacher-action SFT gradient. For $M \in \{\mathrm{OPD}, \mathrm{SFT}\}$, the updated logits are $\theta_M^+ = \theta_0 - \alpha G_M$, and $\pi_M^+ = \mathrm{softmax}(\theta_M^+)$.

The softmax Jacobian is $F = \mathrm{diag}(p) - pp^\top$. At the uniform initial policy, $F = \frac{1}{3}I - \frac{1}{9}\mathbf{1}\mathbf{1}^\top$, hence the first-order probability change under update M is

$$\frac{d}{d\alpha}\pi_M^+\Big|_{\alpha=0} = -FG_M = -\frac{1}{3}G_M.$$

Since the reward vector is $(1, 1, 0)$ and $\sum_i G_{M,i} = 0$, the return derivative equals $G_{M,c}/3$. Therefore

$$\begin{aligned}
\frac{d}{d\alpha}J_{t,h}(\pi_{\mathrm{OPD}}^+)\Big|_{\alpha=0} &= \frac{1}{27}\log\frac{u_a u_b}{u_c^2} = \frac{1}{27}\log\frac{v^2\epsilon(1-\epsilon)}{(1-v)^2} < 0, \\
\frac{d}{d\alpha}J_{t,h}(\pi_{\mathrm{SFT}}^+)\Big|_{\alpha=0} &= \frac{v - 2/3}{3} > 0.
\end{aligned} \tag{11}$$

Let d_{OPD} be the negative of the first derivative and d_{SFT} the second derivative in Eq. 11. Both are strictly positive. Smoothness gives

$$\begin{aligned}
J_{t,h}(\pi_{\mathrm{OPD}}^+) &= J_{t,h}(\pi_S) - d_{\mathrm{OPD}}\alpha + O(\alpha^2), \\
J_{t,h}(\pi_{\mathrm{SFT}}^+) &= J_{t,h}(\pi_S) + d_{\mathrm{SFT}}\alpha + O(\alpha^2).
\end{aligned}$$

Choose $\alpha_0 > 0$ small enough that each remainder has magnitude at most $d_M\alpha/2$ for every $0 < \alpha \leq \alpha_0$. The OPD update then strictly decreases return, while the SFT update strictly increases it, proving Eq. 8.

All probabilities are positive, and the teacher advantage and derivative inequalities are strict. By continuity, these inequalities persist under sufficiently small perturbations of the initial logits and teacher probabilities. The comparison concerns the update direction: both population losses admit the representable teacher as a global minimizer. □

B Mode Seeking and Mode Covering

Figure 6 illustrates the different optimization tendencies of the two KL directions at a fixed interaction history. Reverse KL emphasizes actions already sampled by the student and favors concentrating probability on teacher-supported modes. Forward KL penalizes low student probability on actions supported by the teacher, encouraging coverage of behaviors the student rarely generates. UOPD applies reverse KL to retained student actions and implements forward KL through SFT on teacher corrections. Executing these corrections also changes subsequent interaction histories; this effect on the rollout comes from action replacement, while SFT provides the learning signal.

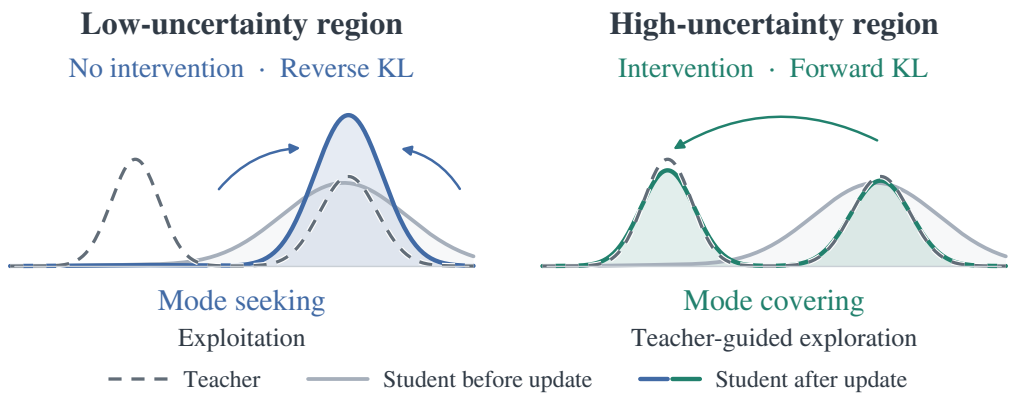


Figure 6: Mode seeking and mode covering in UOPD. Left: reverse KL refines student probability around a teacher-supported mode. Right: forward KL encourages the student to cover teacher behaviors it rarely generates. The curves schematically illustrate the two loss directions rather than measured policy distributions.

C Details of the ALFWorld Motivation Study

This section provides the protocol for the controlled study in Section 3. We compare student-only rollouts, one teacher intervention at a low-confidence step, and one teacher intervention at a random step. Both models remain fixed, so the comparison measures how an action correction changes subsequent student behavior without a parameter update.

C.1 Models, Tasks, and Rollout Collection

We collect student-only rollouts on 100 tasks from the ALFWorld valid-seen split (Shridhar et al., 2021). The student is Qwen2.5-3B-Instruct, evaluated zero-shot, and the teacher is GiGPO-Qwen2.5-7B-Instruct-ALFWorld (Feng et al., 2025). Episodes terminate on task completion or after 50 environment turns. We use sampling temperature 0.4, a maximum of 512 response tokens per turn, and seed 42. Each response contains reasoning and an environment action; the resulting observation is appended to the interaction history. We save the exact prompt and response token IDs and score the recorded student responses under the teacher after rollout collection.

C.2 Teacher Confidence and Intervention Selection

For a student response $a_t^S = (y_{t,1}, \ldots, y_{t,L_t})$ at turn t, we define teacher confidence as the length-normalized log-probability

$$C_t := \frac{1}{L_t} \sum_{j=1}^{L_t} \log \pi_T(y_{t,j} \mid h_t, y_{t,<j}), \tag{12}$$

where h_t is the interaction history defined in Section 4.1. We score all response tokens, including reasoning and the structured action, and exclude prompt tokens. Scores are computed from unscaled teacher logits and reported in nats per token; higher values indicate greater teacher support for the student response. The corresponding uncertainty score is $\delta_t = -C_t$.

We set τ_C to the 25th percentile of valid turn-level confidence scores across the 100 baseline rollouts, giving $\tau_C \approx -1.216$. For trajectory i with H_i recorded turns, the selected intervention turn is

$$t_i^* := \min\{t : 1 \le t < H_i - 1,\ C_{i,t} < \tau_C\}. \tag{13}$$

Turn indices are zero-based. This selects the first eligible threshold crossing, rather than the single lowest-confidence action. We exclude the first and final baseline turns to retain a replay prefix and subsequent student actions for comparison. Of the 100 baseline trajectories, 96 meet this criterion and 93 have verified matching intervention replays. We use these same 93 tasks for all three comparison conditions.

C.3 Matched Single-Turn Interventions

The **Student** condition uses the original student-only trajectory. The **Intervention** condition replaces the action at t_i^* with one teacher-generated action. The **Random** condition instead samples one turn uniformly from $\{1, \ldots, H_i - 2\}$ on the same baseline trajectory, using a task-specific random seed derived from seed 42.

For either intervention condition, we reset the environment to the recorded game file and execute the saved student actions up to the chosen turn. The teacher then generates and executes one action, and the student resumes until task completion or the original 50-turn limit. The intervened rollout therefore shares the baseline prefix and receives exactly one turn of teacher assistance. We match trajectories by game file and reconstructed model inputs, and verify replays against the saved prompts, actions, and final rewards. When Random and Intervention choose the same turn, they share the saved teacher branch; this occurs on four tasks, leaving 275 distinct trajectories across the three conditions.

C.4 Measurements in Figure 2

Teacher confidence. Panel (a) averages C_t over baseline trajectories that reach each displayed turn, using the full set of 100 student-only rollouts. Panel (b) aligns the matched Student and Intervention trajectories at t_i^* and averages teacher confidence on student-generated responses at offsets $1, \ldots, 5$. The teacher's replacement action is excluded from this readout. Each mean uses the trajectories with an available response at that turn; trajectories that have already ended are not padded. We plot turns with at least 15 observations and connect the means without smoothing.

Task success. Panel (c) reports the percentage of the 93 matched tasks completed successfully under each condition, using the environment's terminal success indicator.

State returns. Panel (d) reports the percentage of trajectories that leave a state and later return to it. Let $z_0, \ldots, z_H$ denote the symbolic states before the first action and after each executed action. We identify a state by its exact set of raw PDDL facts, including predicate names, argument identities and types, and simulator bookkeeping facts. Observation text, turn numbers, and interaction history are excluded from this identity. For each trajectory, we compute

$$I_{\text{return}}(\tau) = \mathbf{1}\{\exists\, t \in \{1, \ldots, H\} : z_t \neq z_{t-1} \text{ and } z_t \in \{z_0, \ldots, z_{t-1}\}\}\,.$$

This statistic is computed within each trajectory and covers the full rollout, including the teacher action where present. Consecutive unchanged states do not count as a return, and returning does not require repeating the same action. State returns provide a task-specific indicator of backtracking; a return can also accompany useful information gathering. Table 3 summarizes task success and state returns for all three conditions.

Condition	Task success (%)	Trajectories with state returns (%)
Student	13.98 (13/93)	93.55 (87/93)
Random	16.13 (15/93)	91.40 (85/93)
Intervention	25.81 (24/93)	80.65 (75/93)

Table 3: Outcomes on the same 93 ALFWorld tasks used in Fig. 2(c,d). Random and Intervention each execute one teacher action before returning control to the student.

D Experimental Details

This section describes the benchmarks and data, the student and teacher models, the baselines, and the training and evaluation configurations behind the results in Section 5. Appendix E gives the prompts used for training and evaluation.

D.1 Benchmark Environments

ALFWorld (Shridhar et al., 2021). Text-based household tasks of six types (pick, clean, heat, cool, examine, pick-two) in the TextWorld engine. Training draws from the 3,553 training games. We

evaluate on the 140 seen and 134 unseen validation tasks. A task succeeds when it is completed within 50 environment steps, and an episode's turns are its environment steps, with a failed episode counting the full 50.

WebShop (Yao et al., 2022). Instruction-following product search and purchase over the 1,000-product catalog. Training draws from a pool of 6,410 instructions, and evaluation uses 128 test instructions that are disjoint from the pool and shared by every method. The task score is the environment's attribute-match reward on a 0–100 scale, the success rate is the share of episodes with a perfect match, and the episode cap is 15 steps.

Search (Jin et al., 2025). Open-domain question answering with a retrieval tool, in the Search-R1 setting. Training draws from 169,615 questions (79,168 from NQ and 90,447 from HotpotQA). The test set is Search-R1's four multi-hop datasets, 22,523 questions in total: HotpotQA (7,405), 2WikiMultiHopQA (12,576), MuSiQue (2,417) and Bamboogle (125). Retrieval uses Search-R1's e5-base-v2 encoder over its 2018 Wikipedia corpus with a flat FAISS index, returning the top 3 passages per query. Answers are scored by exact match after normalisation against any gold answer, using the last `<answer>` of the episode. The turn limit is 8.

D.2 Students and Teachers

Benchmark	Students	Teacher
ALFWorld	Qwen2.5-3B-Instruct, Qwen2.5-1.5B-Instruct	GiGPO-Qwen2.5-7B-Instruct-ALFWorld
WebShop	Qwen2.5-3B-Instruct, Qwen2.5-1.5B-Instruct	GiGPO-Qwen2.5-7B-Instruct-WebShop
Search	Qwen2.5-1.5B (base)	SearchR1-qwen2.5-7b-em-ppo

The ALFWorld and WebShop teachers are RL-trained with GiGPO (Feng et al., 2025); the Search teacher is Search-R1's PPO-trained 7B model, trained from a base checkpoint on Search-R1's raw prompt, so the Search student is a base checkpoint prompted the same way. In every pair, teacher and student share a tokenizer, so the teacher can score the student's own tokens.

D.3 Baselines

Zero-shot student and teacher. Evaluated without training under the protocol of Appendix D.5. They bound the results from below and above.

OPD (Agarwal et al., 2024; Lu & Thinking Machines Lab, 2025). The student plays the whole episode, and the teacher scores every student token under the same prompt. Each token receives the advantage $A_t = \lambda \left(\log \pi_T(y_t) - \log \pi_{S,\text{old}}(y_t)\right)$ with $\lambda = 1$, optimised with a PPO objective (clip 0.2, dual clip 3.0, token-mean aggregation). UOPD uses the same distillation term and adds its intervention rule and the SFT loss on intervened turns.

TCOD-F2B and TCOD-B2F (WANG et al., 2026). Temporal curricula over the rollout horizon, advanced every η rollout batches n. F2B lets the student play only the first $W_n = \min(1 + \lfloor n/\eta \rfloor, T_{\max})$ turns of an episode and then ends it. B2F replays a stored expert prefix of $P_n = \text{clamp}(L - 1 - \lfloor n/\eta \rfloor, 0, L - 1)$ actions, where L is the expert trajectory's length, and the student plays from that point on. The prefix carries no loss. Both train the student's turns with the OPD loss and are evaluated on the full horizon. On ALFWorld, η is 6 for F2B and 5 for B2F.

FTB-OPD (Chen et al., 2026). Introduces teacher corrections at steps where the student and teacher disagree most, and keeps a correction for distillation only when it improves the teacher's preference over the student's subsequent continuations. Validating a correction requires paired student continuations, which is the source of its extra training time in Figure 5.

D.4 Training Configuration

We implement UOPD using Trinity-RFT (Pan et al., 2025). We use its asynchronous rollout and training pipeline, with vLLM for student and teacher inference and the verl backend with fully sharded data parallelism (FSDP) for student updates. Table 5 gives the eight-GPU allocation.

For ALFWorld and WebShop, the primary UOPD schedules linearly reduce the target intervention rate from 30% to 5% for the 3B student and from 50% to 30% for the 1.5B student. The intervention threshold is calibrated online using teacher confidence on student-proposed actions. The SFT weight, teacher execution, and intervention schedules are compared in Section 5.3. Table 4 lists the full configuration.

Intervention signals. For a student-proposed action a_t^S at history h_t, the teacher-confidence signal is the teacher's length-normalized negative log-likelihood:

$$\delta_t^{\text{conf}} = -\frac{\log \pi_T(a_t^S \mid h_t)}{|a_t^S|}.$$

The student–teacher confidence gap is

$$\delta_t^{\text{gap}} = \frac{\log \pi_\theta(a_t^S \mid h_t) - \log \pi_T(a_t^S \mid h_t)}{|a_t^S|}.$$

Here $|a_t^S|$ is the number of tokens in the action response, and π_θ is the student policy used to generate the action. Larger scores indicate lower teacher confidence or a larger student–teacher confidence gap, respectively; both trigger intervention when they exceed τ_n.

Table 4: Training hyperparameters. Settings shared by all methods are listed first; the last block is specific to UOPD.

	ALFWorld	WebShop	Search
Optimisation			
Optimizer	AdamW, β=(0.9, 0.999), weight decay 0.01, constant learning rate, no warm-up		
Learning rate	1×10^{-6}	1×10^{-6}	1×10^{-6}
Gradient clipping	1.0	1.0	1.0
Training steps	250	150	300
Tasks per rollout batch	16	16	64
Rollouts per task	1	1	1
Update epochs / mini-batch	1 / 64	1 / 64	1 / 64
Dynamic batching (tokens per GPU)	16,384	16,384	9,728
Sampling	asynchronous, at most 2 policy versions stale		
Loss			
Distillation (reverse-KL) coefficient λ	1.0	1.0	1.0
Rollout			
Max prompt / response tokens	2,048 / 512	4,096 / 512	4,096 / 512
Sampling temperature	1.0	1.0	1.0
Episode limit (steps / turns)	50	15	8
History in the prompt	last 2 steps	last 2 steps	full episode
Retrieval	–	–	top 3 passages
UOPD			
Intervention signal	Teacher confidence	Teacher confidence	Confidence gap
Scoring temperature for the signal	1.0	1.0	1.0
Intervention schedule, 3B	$30\% \to 5\%$ over 120	$30\% \to 5\%$ over 75	–
Intervention schedule, 1.5B	$50\% \to 30\%$ over 120	$50\% \to 30\%$ over 75	constant 30%
SFT weight β, form	1.0	1.0	1.0

Table 5: Eight-GPU allocation. Student rollout, teacher inference, and student updates run on separate GPU groups.

Role	GPUs	Configuration
Student rollout (vLLM)	4	2 engines, tensor parallel size 2 per engine
Teacher inference (vLLM)	2	1 engine, tensor parallel size 2
Student updates (verl/FSDP)	2	Ulysses sequence parallel size 2
Total	8	Single node

D.5 Evaluation Protocol

ALFWorld. We evaluate household task completion on 140 seen and 134 unseen tasks, reporting the percentage of successfully completed tasks in each split. Average turns are computed separately for each split over all episodes, including failures.

WebShop. We use the 1000-product catalog and 128 test tasks. Task score measures how well the selected product satisfies the request; success rate measures full task completion. Average turns include both successful and failed episodes. For ALFWorld and WebShop, Table 1 reports means and standard deviations over three evaluation seeds.

Search. Figure 3 reports exact match on each of the four multi-hop test sets of Search-R1 (Jin et al., 2025): HotpotQA, 2WikiMultiHopQA, MuSiQue and Bamboogle, 22,523 questions in total. Average turns are computed over the same 22,523 questions. A turn is one model response, the answering one included, so an episode that reaches the limit and receives the forced answer counts nine turns. Both OPD and UOPD use seed 42, greedy decoding, forced answers, and an eight-turn limit.

	ALFWorld	WebShop	Search
Temperature	0.4	0.4	0 (greedy)
Max prompt / response tokens	2,048 / 512	4,096 / 512	as in training / 512
Episode limit	50	15	8, then a forced answer
Test set	140 seen, 134 unseen	128 test	22,523 multi-hop questions

Every evaluation uses the checkpoint of the final training step. A Search episode that reaches the turn limit without an answer is given one additional turn that asks for the final answer (Appendix E.3); this turn is used only at evaluation.

D.6 Hardware

Training runs use a single node, primarily with eight NVIDIA RTX PRO 6000 Blackwell GPUs. We also use four-GPU setups with NVIDIA H100 (80 GB) or NVIDIA RTX A6000 (48 GB) GPUs. All reported training-time comparisons use the eight-GPU RTX PRO 6000 Blackwell setup, with the allocation in Table 5.

E Environment Prompts

This section gives the prompts used for each environment during training and evaluation, verbatim. Braces mark the fields filled at each step. The student and the teacher receive the identical prompt, which lets the teacher score the student's own tokens, and each prompt is sent as a single chat-formatted user message. We pass no system message, so the model's chat template adds its default one: "You are Qwen, created by Alibaba Cloud. You are a helpful assistant." for the Qwen2.5-Instruct students, and "You are a helpful assistant." for the base Search student. No prompt states the step or turn limit; the environment enforces it.

E.1 ALFWorld Prompts

ALFWorld is a text-based household environment in which the agent completes object-manipulation tasks. At each step the agent reasons inside `<think> </think>` tags and then gives one admissible action inside `<action> </action>` tags, which is executed in the environment. The first step has its own template without the task, which the first observation already contains; every later step uses the second template.

ALFWorld Prompt Template (first step)

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

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

ALFWorld Prompt Template (later steps)

```
You are an expert agent operating in the ALFRED Embodied Environment. Your task is to:
     {task_description}
Prior to this step, you have already taken {step_count} step(s). Below are the most recent
     {history_length} observations and the corresponding actions you took: {action_history}
You are now at step {current_step} and your current observation is: {current_observation}
Your admissible actions of the current situation are: [{admissible_actions}].

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

{admissible_actions} lists the environment's admissible commands except help, each in single quotes, one per line. The history holds the most recent steps, one per line, each as [Observation {i}: '{obs}', Action {i}: '{act}'], and {history_length} is the number shown: $\min(2, \text{steps taken})$.

E.2 WebShop Prompts

WebShop is a simulated e-commerce website in which the agent searches for, selects and buys a product that matches a natural-language instruction. The first template is used for the first two steps and the second from the third step on, always with the two most recent steps as history ($\{\texttt{history_length}\} = 2$, entries as in ALFWorld, each observation flattened to one line).

WebShop Prompt Template (first two steps)

```
You are an expert autonomous agent operating in the WebShop e-commerce environment.
Your task is to: {task_description}.
Your current observation is: {current_observation}.
Your admissible actions of the current situation are:
[
{available_actions}
].

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

WebShop Prompt Template (later steps)

```
You are an expert autonomous agent operating in the WebShop e-commerce environment.
Your task is to: {task_description}.
Prior to this step, you have already taken {step_count} step(s). Below are the most recent
     {history_length} observations and the corresponding actions you took: {action_history}
You are now at step {current_step} and your current observation is: {current_observation}.
Your admissible actions of the current situation are:
[
{available_actions}
].

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

Action format for WebShop. {available_actions} lists one action per line, of two types:

- search[<query>]: search the catalog with a text query; listed only when the page has a search bar.
- click[<element>]: click an element of the current page, such as a product, an option or a navigation button; one line per clickable element.

E.3 SEARCH PROMPTS

Search is open-domain question answering with a retrieval tool. The prompt is Search-R1's original one, on which the teacher was trained, reproduced verbatim.

Search Prompt Template (Search-R1's original prompt)

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

The episode so far is appended directly after the prompt: each earlier model output, cut after its first complete <search>...</search> or <answer>...</answer>, followed by the environment's reply. After a search the reply is the top 3 passages, on its own line. An output with neither a complete <search> nor a complete <answer> receives an invalid-action reply instead.

Retrieved Passages (appended after each search)

```
<information>Doc 1: {passage}
Doc 2: {passage}
Doc 3: {passage}</information>
```

Invalid-Action Reply

```
Invalid action. After your reasoning, choose exactly one action: <search> your query
    </search> or <answer> your final answer </answer>.
```

Generation stops at </search> or </answer>. A search continues the episode and an answer ends it; the episode is scored by exact match of its last <answer>. As in Search-R1, the prompt does not state the turn limit; the environment ends the episode after 8 turns. In training, an episode that reaches the limit without an answer scores 0. At evaluation it receives one more turn, whose prompt is the Search prompt above with the episode so far, followed on a new line by the forced-answer instruction.

Forced Final Answer (evaluation only)

```
You can no longer search. Based on the information above you MUST now give your final
    answer inside <answer> and </answer>. For example, <answer> Beijing </answer>.
```

F INTERVENTION DYNAMICS DURING TRAINING

We show intervention dynamics on ALFWorld for the 3B student with a $30\% \to 5\%$ schedule, examining how often UOPD intervenes and where corrections occur during training. The run uses teacher confidence, $\beta = 1$, and teacher action execution. The curves show unsmoothed batch statistics from this training run. The horizontal axis is the rollout batch index n in Eq. 2; asynchronous rollout collection and optimization use different counters. The schedule decays over 120 rollout batches.

The actual intervention rate is computed within each trajectory and then averaged across the batch. For intervention position, we average the zero-based turn indices of corrections within each trajectory, then average over trajectories with at least one correction. Trajectories without corrections contribute zero to the rate but are excluded from the position mean.

Figure 7 shows a decreasing intervention rate. The episode-averaged rate falls below the scheduled target, particularly late in training. Part of this gap comes from how trajectories are weighted: quantile calibration uses a window of action scores, whereas the plotted rate gives each trajectory equal weight. Over the final 20 batches, the episode-averaged rate is approximately 2%, while averaging the batch-level ratios of total corrections to total turns gives 4%, closer to the 5% target. This difference indicates that corrections are more concentrated in longer trajectories.

Mean intervention position remains variable and does not show a sustained shift toward earlier turns. Only about 22% of trajectories receive any correction in the final 20 batches, so the position

curve describes a small subset of the collected rollouts. Together, these results show that declining intervention frequency need not imply a uniform shift in correction position: UOPD continues to select turns according to uncertainty within the trajectories encountered by the student.

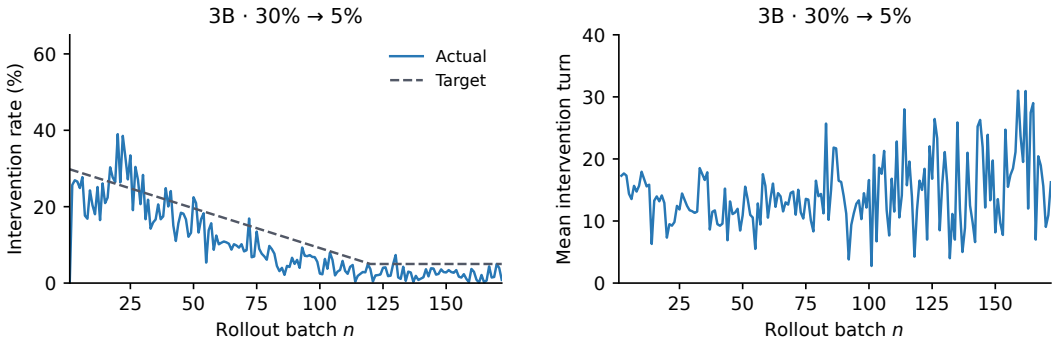


Figure 7: Intervention dynamics on ALFWorld with a 3B student and a $30\% \to 5\%$ schedule. Left: actual episode-averaged intervention rate (blue) and scheduled target (dashed gray). Right: mean intervention turn among corrected trajectories. Interventions become less frequent, while their positions remain variable.