INSIDE FAILED ROLLOUTS: PROGRESSIVE EXPERIENCE SELF-DISTILLATION FOR AGENTIC TRAINING

Anonymous authors
Paper under double-blind review

ABSTRACT

On-policy self-distillation complements sparse outcome rewards with dense supervision by re-scoring an agent's own rollouts under privileged training-time context. Failed rollouts are particularly challenging because useful progress and harmful decisions often coexist within the same trajectory. We introduce Progressive Experience Self-Distillation (PESD), a failure-aware framework that uses privileged likelihood shifts to separate realized actions into supported, rejected, and uncertain regions. Supported actions receive forward distillation, consistently rejected actions receive low-weight unlikelihood updates, and uncertain actions are left to RL. PESD further organizes outcome feedback, applicability-gated skills, and verified successful trajectories into a progressive privilege hierarchy, with stronger guidance reduced across repeated task visits and removed at inference. On ALFWorld, PESD reaches 92.2/66.1 with Qwen2.5-3B-Instruct/Qwen3-1.7B, and 46.0/44.0 on Search-QA. These results show that failed rollouts can provide useful action-level supervision when privileged signals are applied selectively rather than uniformly.

1 INTRODUCTION

Large language models are increasingly trained as agents that interact with external environments over multiple turns (Luo et al., 2025a; Xi et al., 2025; Liu et al., 2024; Schick et al., 2023; Patil et al., 2024; Shi et al., 2025; Wang et al., 2023). In these agentic settings (Shridhar et al., 2020; Luo et al., 2025b; Wang et al., 2025; Wu et al., 2026b; Zhang et al., 2026; Yan et al., 2026; Li et al., 2026b), agents must make a sequence of dependent decisions about observations, tool calls, and environment states, and useful feedback often arrives only after several interaction steps. Reinforcement learning (RL) (Jin et al., 2025; Shao et al., 2024; Guo et al., 2025; Yu et al., 2026; Feng et al., 2026; Dong et al., 2026; Gehring et al., 2024; Qi et al., 2025; Qian et al., 2026) is a standard choice for optimizing such multi-turn behavior. However, the feedback available to RL is often sparse and delayed: a trajectory may receive only a single outcome-level reward (Andrychowicz et al., 2017; Arjona-Medina et al., 2019). When the trajectory fails, this reward identifies the final outcome but gives little guidance about the individual decisions that led there. This is problematic because failed on-policy rollouts are rarely uniformly bad; they often contain useful intermediate progress as well as actions that derail the task (Shinn et al., 2023; Zhao et al., 2024; Madaan et al., 2023; Ge et al., 2025). The key challenge is therefore not whether failed experience should be used, but how to decide which parts should be preserved and which should be corrected.

On-policy distillation (OPD) complements trajectory-level rewards with dense teacher supervision on trajectories sampled from the student policy (Hinton et al., 2015; Kim & Rush, 2016; Agarwal et al., 2024; Gu et al., 2024; Ye et al., 2026; Yang et al., 2026c). On-policy self-distillation (OPSD) (Zhao et al., 2026) removes the need for a separate teacher by evaluating the student's own rollouts under asymmetric contexts. The student acts from the original interaction history, while a privileged branch re-scores the same actions with additional training-time information. The resulting likelihood shifts provide dense feedback on realized on-policy behavior, but their reliability depends on the privileged context and can vary across decisions within the same trajectory. Recent methods (Wang et al., 2026a; Lu et al., 2026a; Wang et al., 2026b; Lu et al., 2026b; Ding, 2026; Li et al., 2026a) improve robustness by gating or attenuating unfavorable teacher discrepancies. This helps suppress unreliable signals, but it does not address how negative privileged signals should

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.

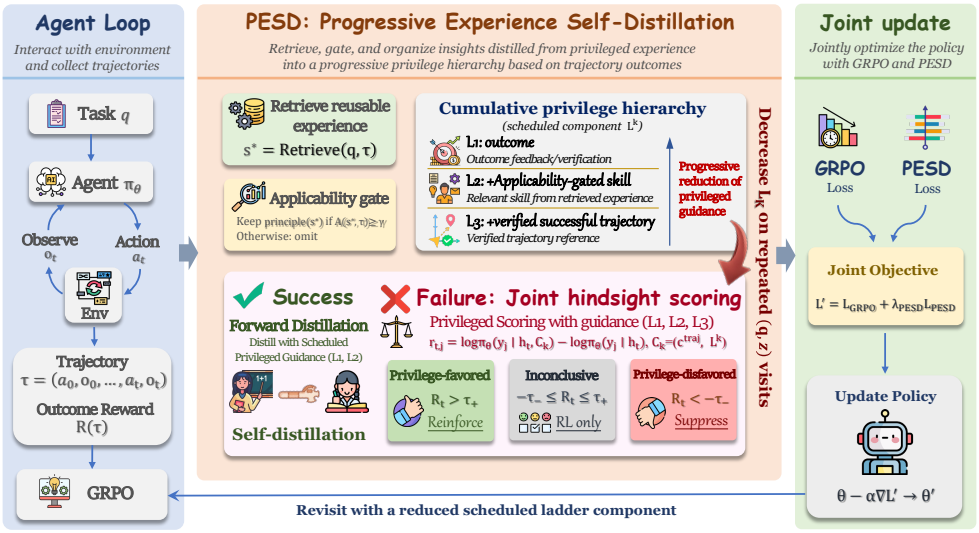


Figure 1: Overview of PESD. Policy snapshot re-scores actions from a failed rollout under $C_k = (c^{\text{traj}}, L_k)$, producing the $\mathcal{P}/\mathcal{N}/\mathcal{U}$ partition.

be used once the trajectory itself has failed. It also leaves open how the privileged context should change as training progresses.

The difficulty is especially clear when the on-policy trajectory fails. A failed episode is rarely uniformly wrong: it may include useful observations, appropriate tool use, and several correct intermediate decisions before a small number of actions derail the outcome (Andrychowicz et al., 2017; Arjona-Medina et al., 2019; Uesato et al., 2022; Madaan et al., 2023; Ge et al., 2025; Wu et al., 2026a; Jiang et al., 2024). Penalizing the whole trajectory would discard this useful progress, while simply filtering out negative teacher shifts would leave corrective information unused. We introduce Progressive Experience Self-Distillation (PESD), a failure-aware approach to self-distilled agent training that uses privileged hindsight to decide how realized actions should be updated. For each failed trajectory, PESD constructs a teacher context from the completed episode and a progressive hierarchy of outcome feedback, applicability-gated skills, and verified successful trajectories. A fixed policy snapshot then re-scores the student's realized actions under this context, while the student itself is trained only from the original interaction history. Actions supported by the privileged context receive forward distillation; actions with large and sign-consistent negative shifts receive low-weight unlikelihood supervision; and actions with inconclusive evidence are left to RL. This selective treatment preserves useful behavior in failed rollouts while restricting correction to actions with clearer negative evidence.

The privileged guidance is also scheduled over repeated attempts. For each task–outcome pair, PESD tracks an active privilege level and lowers it when the same branch is revisited. Each revisit uses a fresh rollout from the updated policy, rather than replaying stored trajectories. The completed episode remains available for hindsight scoring, while the additional guidance is gradually reduced from verified successful trajectories to gated skills and outcome feedback. This schedule encourages the policy to first learn with stronger hindsight and then retain the behavior under weaker forms of guidance. We evaluate PESD on ALFWorld and Search-QA with Qwen2.5-3B-Instruct (Yang et al., 2025b) and Qwen3-1.7B (Yang et al., 2025a). PESD reaches 92.2 and 66.1 on ALFWorld, and 46.0 and 44.0 on Search-QA, respectively. Our main contributions are:

- We introduce failure-aware self-distillation for failed on-policy trajectories, using privileged likelihood shifts to decide which actions should be reinforced, suppressed, or left to RL.
- We develop a progressive privilege hierarchy that organizes outcome feedback, retrieved skills, and verified successful rollouts into increasingly informative levels, then reduces this guidance across task revisits.
- We evaluate PESD on ALFWorld and Search-QA across two model scales, showing consistent gains over RL and self-distillation baselines. Ablations and action-level analysis further support the design of PESD.

2 Related Work

2.1 Agentic Reinforcement Learning.

Reinforcement learning has become a common post-training approach for language-model reasoning and decision making (Shao et al., 2024; Guo et al., 2025; Yu et al., 2026), and has been extended to multi-turn agents in embodied, web, search, and software environments (Liu et al., 2024; Wang et al., 2023; Shridhar et al., 2020; Jin et al., 2025; Wang et al., 2025). A central challenge in these settings is sparse or delayed feedback. A trajectory-level reward can indicate whether the agent succeeds, but it provides limited information about which intermediate decisions were responsible for the outcome. Prior work has explored hindsight relabeling, return decomposition, process supervision, and reflection-based feedback to improve credit assignment over long trajectories (Andrychowicz et al., 2017; Arjona-Medina et al., 2019; Lightman et al., 2024; Uesato et al., 2022; Shinn et al., 2023; Madaan et al., 2023; Zhao et al., 2024). Our work studies a complementary problem: how to extract action-level supervision from failed on-policy trajectories, where useful progress and harmful decisions often coexist in the same rollout.

2.2 On-Policy Distillation and Self-Distillation.

On-policy distillation (OPD) (Agarwal et al., 2024; Ye et al., 2026; Yang et al., 2026c) trains on trajectories sampled from the student policy while obtaining dense supervision from a teacher, reducing the distribution mismatch of offline distillation. On-policy self-distillation (OPSD) (Zhao et al., 2026) removes the external teacher by evaluating the same model under ordinary and privileged contexts. This idea has been combined with reinforcement learning and extended to multi-turn agents in recent work (Yang et al., 2026a; Wang et al., 2026a; Ding, 2026; Wang et al., 2026b; Zhong et al., 2026; Yang et al., 2026b). SDAR (Lu et al., 2026a) improves stability by gating privileged supervision and attenuating unfavorable discrepancies, while PCSD (Lv et al., 2026) further filters teacher signals using local persistence. These methods improve the reliability of privileged supervision, but they do not directly address how negative privileged evidence should be used when the rollout itself has failed. PESD focuses on this setting. It uses consistently rejected actions as conservative corrective supervision, while leaving ambiguous actions to RL.

2.3 Experience and Skill Learning for Agents.

Agents can also improve by turning prior interactions into reusable knowledge, such as reflections, episodic experience, or higher-level skills (Tang et al., 2025; Zhao et al., 2024; Shinn et al., 2023; Madaan et al., 2023). SkillRL (Xia et al., 2026) organizes experience from successful and failed trajectories into a SkillBank, while Skill-SD (Wang et al., 2026a) and OPID (Yang et al., 2026b) use retrieved skills as privileged context for self-distillation. Evolving Rollouts (Wang et al., 2026c) and SEED (Wu et al., 2026a) further derive guidance from reward-labeled or evolving on-policy experience. PESD differs in how this experience is used for supervision. It filters retrieved skills by applicability, combines them with completed-episode hindsight and verified successful trajectories, and places them in a privilege hierarchy that is gradually reduced across repeated visits. The resulting experience is used only to score the agent's realized on-policy behavior during training and is not required at inference.

3 Method

3.1 Problem Setup

Given a task q, the current policy π_θ interacts with the environment and generates an on-policy trajectory $\tau = (o_0, a_0, \ldots, o_T, a_T)$ with outcome reward $R(\tau)$. Reinforcement learning optimizes task-level performance from this trajectory-level signal, but provides little information about individual decisions. This is especially limiting for failed trajectories, which often contain useful progress alongside decisions that derail the outcome. On-policy self-distillation provides denser supervision by re-scoring the student's own behavior with information available only during training. The student acts from the original interaction history, while a privileged branch evaluates the same realized actions with additional hindsight. For failed trajectories, PESD uses the resulting likeli-

hood shifts to partition actions into three supervision regions: $\mathcal{P}$ for privileged-supported actions, $\mathcal{N}$ for consistently rejected actions, and $\mathcal{U}$ for inconclusive actions. These regions receive forward distillation, low-weight unlikelihood supervision, or no auxiliary update, respectively. This allows PESD to preserve useful behavior, suppress rejected behavior, and leave ambiguous decisions to RL without treating the failed rollout uniformly. Successful trajectories are handled separately through forward self-distillation, since their behavior is already validated by the environment. All privileged information is used only during training; inference proceeds from the original interaction context.

3.2 Privileged Experience

To construct the privileged signal used for this action-level supervision, PESD combines failure-conditioned retrieval with a structured teacher context.

Failure-conditioned retrieval. We maintain a repository of reusable experience $\mathcal{B} = \{s_1, \ldots, s_N\}$ and use the SkillBank retrieval mechanism of SkillRL (Xia et al., 2026). Because task-level similarity does not necessarily capture the cause of a particular failure, retrieval is conditioned on the current rollout:

$$s^* = \mathrm{Retrieve}(q, \tau). \tag{1}$$

Different failures of the same task may therefore retrieve different reusable experience.

Applicability-gated skills. Retrieved skills encode reusable principles rather than instance-specific instructions. A lightweight gate determines whether the retrieved principle should enter the teacher context:

$$E^{\mathrm{skill}} = \begin{cases} \{\mathrm{principle}(s^*)\}, & A(s^*, \tau) \geq \gamma, \\ \varnothing, & \text{otherwise.} \end{cases} \tag{2}$$

where $A(s^*, \tau)$ measures the applicability of the retrieved skill to the current trajectory. The gate therefore filters retrieved experience before it enters the privilege hierarchy. Implementation details are provided in Appendix C.2.

Privileged context. PESD re-scores each on-policy trajectory under the teacher context $C_k = (c^{\mathrm{traj}}, L_k)$, which combines completed-episode hindsight with the additional guidance available at privilege level k. For a failed rollout, c^{traj} is the student's completed trajectory together with its unsuccessful outcome; for a successful rollout, it is a verified success from the same group. The context is used only for teacher-side scoring, while the student continues to act from the original interaction history.

The additional guidance is organized into a cumulative hierarchy, $L_1 \subseteq L_2 \subseteq L_3$:

$$\begin{aligned} L_1 &= \{e^{\mathrm{out}}\}, \\ L_2 &= L_1 \cup E^{\mathrm{skill}}, \\ L_3 &= L_2 \cup \{\tau^{\mathrm{ref}}\}. \end{aligned} \tag{3}$$

where e^{out} denotes outcome feedback and e^{skill} is the applicability-gated retrieved skill. The reference trajectory τ^{ref} is selected from successful on-policy rollouts produced during training and verified by the environment. When multiple successful rollouts are available for the same task, we choose the shortest one as the reference. Only trajectories satisfying $V(\tau^{\mathrm{ref}}) = 1$ are eligible; if no verified success is available, the hierarchy starts from L_2. The reference is used only as privileged context for scoring and is never treated as a demonstration target.

3.3 Progressive Privilege Schedule

Strong privilege helps expose corrective structure early, but keeping the same level of guidance throughout training can make the policy depend on information that will not be available at inference time. PESD therefore maintains an independent privilege state for each task–outcome pair (q, z), where $z \in \{\mathrm{success}, \mathrm{failure}\}$, and reduces the active level across repeated visits. A newly observed success starts from L_2, while a newly observed failure starts from L_3 when a verified reference is available and from L_2 otherwise. Each subsequent visit to the same task–outcome pair decreases the level by one, bounded below by L_1, i.e., $k_{\mathrm{new}} = \max(1, k_{\mathrm{prev}} - 1)$. Success and failure histories are tracked separately.

Each revisit generates a fresh rollout from the latest policy rather than replaying stored trajectories. Previously encountered tasks are mixed with new tasks through a revisit buffer, and retrieval, applicability gating, and teacher-side scoring are recomputed from the new trajectory. The completed-episode context remains available throughout, while the additional guidance is progressively reduced from a verified trajectory to a gated skill and finally to outcome feedback alone.

3.4 Failure-Aware Distillation

Failed trajectories should not be distilled uniformly. PESD instead uses privileged hindsight to determine which realized actions should be reinforced, suppressed, or left to RL.

Privileged scoring. Let θ_{old} denote the policy that generated the current batch, and let $\bar{\theta} = \text{sg}(\theta_{\text{old}})$ be its stop-gradient snapshot. For token $y_{t,j}$ in action a_t, PESD measures the change induced by the privileged context as

$$r_{t,j} = \log \pi_{\bar{\theta}}(y_{t,j} \mid h_{t,j}, C_k) - \log \pi_{\bar{\theta}}(y_{t,j} \mid h_{t,j}), \tag{4}$$

with both terms detached from optimization. Since the two forward evaluations use the same policy snapshot and differ only in C_k, $r_{t,j}$ captures how hindsight changes the model's preference for the realized token.

For each environment-facing action span $\mathcal{A}_t$, we average the token-level shifts as $R_t = \frac{1}{|\mathcal{A}_t|} \sum_{j \in \mathcal{A}_t} r_{t,j}$. A negative average alone is not sufficient for reverse supervision, since it may be driven by a small subset of tokens. We therefore also measure whether the negative signal is consistent across tokens whose likelihood changes under the privileged context. Let $\mathcal{S}_t = \{j \in \mathcal{A}_t : |r_{t,j}| > 0\}$ and $c_t^- = \frac{1}{|\mathcal{S}_t|} \sum_{j \in \mathcal{S}_t} \mathbb{I}[r_{t,j} < 0]$, with $c_t^- = 0$ when $\mathcal{S}_t$ is empty. Here τ_- and ρ jointly control how selectively failed actions enter the reverse branch, while τ_+ governs admission to the positive branch. PESD then partitions each action as

$$\text{region}(a_t) = \begin{cases} \mathcal{P}, & R_t > \tau_+, \\ \mathcal{N}, & R_t < -\tau_- \text{ and } c_t^- \geq \rho, \\ \mathcal{U}, & \text{otherwise.} \end{cases} \tag{5}$$

Actions in $\mathcal{P}$ are supported by the privileged context and receive forward distillation. Actions in $\mathcal{N}$ require both a sufficiently negative shift and consistent rejection across the action span, and receive conservative suppression. All remaining actions fall into $\mathcal{U}$, receive no auxiliary update, and remain optimized only through RL. The success branch consolidates behavior already validated by the environment, whereas the failure branch decomposes mixed-quality rollouts into actions to preserve, suppress, or defer.

Signed distillation. The resulting failure-aware objective is

$$\begin{aligned} \mathcal{L}_{\text{signed}} = -\sum_t \sum_{j \in \mathcal{A}_t} \Big[& \mathbb{I}[a_t \in \mathcal{P}]\, w_{t,j}^+ \log \pi_\theta(y_{t,j} \mid h_{t,j}) \\ & + \lambda_{\text{rev}}\, \mathbb{I}[a_t \in \mathcal{N}]\, w_{t,j}^- \log(1 - \pi_\theta(y_{t,j} \mid h_{t,j})) \Big]. \end{aligned} \tag{6}$$

Here π_θ is the trainable student, while the detached snapshot $\pi_{\bar{\theta}}$ is used only for scoring. Region assignments and token weights are computed from this snapshot and held fixed during the policy update. Positive weights $w_{t,j}^+$ follow the persistence-based weighting of PCSD (Lv et al., 2026), while $w_{t,j}^-$ is proportional to $[-r_{t,j}]_+$ and renormalized so that the mean weight over $\{u \in A_a : r_{t,u} < 0\}$ is one. This suppression term lowers the probability of the observed realization without constructing an alternative action, and its contribution is kept small through λ_{rev}.

Failure-side distillation is activated only after an initial warmup and then introduced gradually through the coefficient α_u, which scales both the positive and reverse branches. Full weighting and warmup details are provided in Appendix F.

Successful trajectories. Successful trajectories use forward self-distillation only. We use a frozen reference policy π_{ref}, initialized from the model before training, to score the student's successful

tokens under C_k. Tokens that are positively supported by this reference policy are distilled using the persistence-based weighting. A first-visit success uses $C_2 = (c^{\text{traj}}, L_2)$, rather than the full L_3 context. No reverse supervision is applied.

The two branches serve complementary purposes. The success branch consolidates behavior already validated by the environment. The failure branch instead decomposes mixed-quality rollouts into actions to preserve, suppress, or defer. Together, they allow PESD to learn from both successful and failed on-policy experience without replacing student behavior with demonstrations.

On-policy targets. All distillation targets come from the student's own on-policy trajectories. Retrieved skills and verified trajectories are used only to construct the teacher context for scoring these realized actions; they do not provide replacement actions or synthetic continuations. Thus, PESD uses privileged experience to shape supervision over on-policy behavior while keeping the training targets on-policy.

Training objective. We combine PESD with the trajectory-level GRPO objective:

$$\mathcal{L} = \mathcal{L}_{\text{GRPO}} + \lambda_{\text{PESD}} \mathcal{L}_{\text{PESD}}, \tag{7}$$

where

$$\mathcal{L}_{\text{PESD}} = \sum_{\tau} \left[\mathbb{I}[R(\tau) = 1] \, \mathcal{L}_{\text{fwd}}(\tau) + \alpha_u \, \mathbb{I}[R(\tau) = 0] \, \mathcal{L}_{\text{signed}}(\tau) \right]. \tag{8}$$

Successful trajectories are consolidated through forward distillation, while failed trajectories receive the selective signed supervision defined above. GRPO optimizes trajectory-level outcomes, while PESD provides complementary action-level supervision on the same on-policy experience. At inference time, the policy acts from the original interaction context without completed episodes, retrieved skills, or verified trajectories.

4 Experiments

4.1 Experimental Setup

Benchmarks and models. We evaluate PESD on ALFWorld (Shridhar et al., 2020) and Search-QA (Jin et al., 2025), two sequential decision-making settings with distinct action interfaces and sparse outcome feedback. ALFWorld casts household tasks as text-based embodied interaction, requiring an agent to interpret observations and execute multi-step action sequences. We consider six task families: Pick, Look, Clean, Heat, Cool, and Pick2. Search-QA evaluates tool-augmented information seeking, where an agent iteratively issues search queries, gathers evidence, and answers questions drawn from a mixture of single-hop and multi-hop datasets: Natural Question (NQ) (Kwiatkowski et al., 2019), TriviaQA (Joshi et al., 2017), PopQA (Mallen et al., 2023), HotpotQA (Yang et al., 2018), 2WikiMultiHopQA (Ho et al., 2020), MuSiQue (Trivedi et al., 2022), and Bamboogle (Press et al., 2023). We use Qwen2.5-3B-Instruct (Yang et al., 2025b) and Qwen3-1.7B (Yang et al., 2025a) as backbones.

Baselines. We compare PESD with outcome-based RL and representative on-policy distillation methods. *Vanilla* evaluates the instruction-tuned backbone without post-training, while GRPO optimizes trajectory-level rewards without auxiliary distillation. GRPO+OPSD (denoted G-OPSD) combines privileged self-distillation with reinforcement learning. Skill-SD (Wang et al., 2026a) further introduces reusable skills as privileged context for multi-turn self-distillation. We also include RLSD, SDAR (Lu et al., 2026a), and PCSD (Lv et al., 2026), which modify how privileged supervision is integrated or filtered during RL. We follow the original training recipe for the baselines above. Controlled ablations in Section 5.1 isolate the contribution of individual PESD components.

Implementation. We use GRPO as the RL backbone and adopt the SkillBank retrieval mechanism from SkillRL (Xia et al., 2026). Failed trajectories are scored with a stop-gradient copy of the policy that generated the current batch, with $\tau_+ = 0.05$, $\tau_- = 0.20$, and $\rho = 0.60$ throughout. For Search-QA, we follow the Search-R1 (Jin et al., 2025) setup and use E5 (Wang et al., 2022) as the retriever; training uses NQ and HotpotQA, while the remaining datasets are evaluated out of domain. For ALFWorld, we use the GiGPO (Feng et al., 2026) split with 16 tasks per batch and 8 rollouts per task, yielding a batch size of 128, and set the maximum prompt length to 2,048 tokens. All experiments are conducted on four NVIDIA H20 GPUs. Additional implementation details are provided in Appendix F.

4.2 Main Results

ALFWorld. PESD achieves the best overall performance with both backbones, reaching 92.2 with Qwen2.5-3B-Instruct and 66.1 with Qwen3-1.7B. Compared with the strongest baseline, PCSD, this corresponds to gains of 1.6 and 6.7 points, respectively, with the larger improvement observed on Qwen3-1.7B. The results suggest separating actions to preserve from actions to suppress provides a clearer learning signal than treating the failed trajectory as a single negative outcome. Overall, PESD consistently improves over existing RL and self-distillation baselines across both model scales.

Table 1: Performance on ALFWorld. We report success rates (%) for each task category and pooled success over all evaluation tasks. PESD results are averaged over three runs. Best and second-best results are shown in **bold** and underlined, respectively.

	Qwen2.5-3B-Instruct							Qwen3-1.7B						
Method	**Pick**	**Look**	**Clean**	**Heat**	**Cool**	**Pick2**	**Avg**	**Pick**	**Look**	**Clean**	**Heat**	**Cool**	**Pick2**	**Avg**
Vanilla	44.4	11.1	6.2	15.4	28.6	12.5	21.9	25.0	22.2	3.1	0.0	21.4	4.2	12.5
GRPO	91.2	62.5	96.2	61.9	65.0	47.4	75.0	71.1	41.7	36.4	40.0	31.8	31.6	46.1
G-OPSD	**100.0**	82.4	85.7	75.0	70.0	60.0	81.2	38.2	50.0	30.8	28.6	30.0	21.1	32.0
Skill-SD	88.2	50.0	96.2	52.4	65.0	57.9	73.4	52.9	37.5	69.2	42.9	60.0	36.8	52.3
RLSD	87.9	75.0	90.9	75.0	73.1	68.4	79.7	50.0	37.5	61.5	19.0	50.0	21.1	42.2
SDAR	97.1	62.5	**100.0**	61.9	75.0	84.2	84.4	73.5	25.0	**76.9**	33.3	40.0	36.8	53.9
PCSD	**100.0**	63.6	82.1	83.3	**94.4**	**100.0**	90.6	63.3	**68.8**	50.0	69.2	48.1	**63.6**	59.4
PESD	99.0	**86.1**	94.9	**97.8**	84.0	87.0	**92.2**	**79.2**	41.7	66.7	**71.1**	**73.3**	44.5	**66.1**

Search-QA. Following the evaluation protocol in SDAR (Lu et al., 2026a), we report results across seven Search-QA datasets in Table 2. PESD achieves the best overall average with both backbones, reaching **46.0** with Qwen2.5-3B-Instruct and **44.0** with Qwen3-1.7B, compared with 44.6 and 42.2 for the strongest competing results, respectively. The gains are not concentrated on a single dataset: PESD remains strong across both single-hop and multi-hop search tasks, and with Qwen3-1.7B it obtains the highest score on all seven datasets. This suggests that failure-aware supervision improves search-based agents across different query and evidence-gathering patterns, rather than depending on a particular dataset structure. Sample-level aggregation is reported in Appendix A.

4.3 Unseen-Task Evaluation and Training Dynamics

Unseen-task evaluation. We further evaluate PESD with Qwen2.5-3B-Instruct on all 134 tasks from the ALFWorld `valid_unseen` split. As shown in Figure 2, PESD achieves an average success rate of **89.6**, outperforming PCSD (Lv et al., 2026) and SDAR (Lu et al., 2026a). The improvement spans multiple task families, with success rates of 88.9 on Look, 93.5 on Clean, and 95.2 on Cool. These results indicate that the learned failure-aware supervision transfers beyond the training instances, rather than only improving performance on seen tasks.

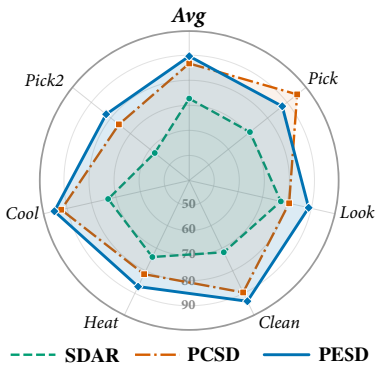


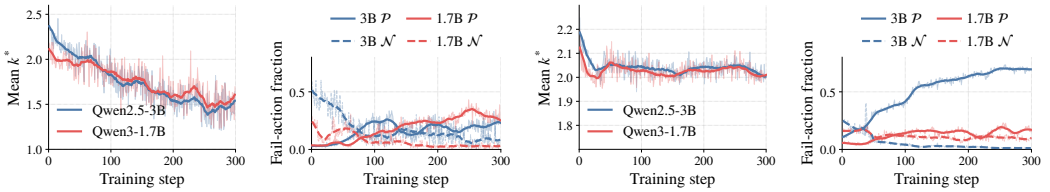
Figure 2: Evaluation on `valid_unseen`.

Privilege and failure dynamics. Figure 3 tracks the nominal ladder level and failure-action partition throughout training. On ALFWorld, the mean level decreases from about 2.5 to 1.6 as task–outcome branches are revisited. Meanwhile, the fraction of failed actions assigned to $\mathcal{N}$ drops from 0.59 to 0.09, while the $\mathcal{P}$ fraction increases from 0.03 to 0.22. Most failed actions remain in $\mathcal{U}$, indicating that reverse supervision is applied selectively rather than broadly penalizing failed rollouts.

On Search-QA, the level changes more gradually, from about 2.2 to 2.0. The failure partition still shifts toward more privileged-supported actions, while $\mathcal{N}$ assignments remain sparse after warmup. These trends are consistent with the intended role of the reverse branch: it provides corrective pressure when negative evidence is clear, but remains conservative throughout training.

Table 2: Search-QA accuracy (%) under the slice-averaged protocol used by prior work. Best and second-best results under this protocol are shown in **bold** and underlined, respectively. Sample-level aggregation is reported in Appendix A.

	Qwen2.5-3B-Instruct								Qwen3-1.7B							
Method	**NQ**	**Triv**	**Pop**	**Hotp**	**2Wk**	**MuS**	**Bam**	**Avg**	**NQ**	**Triv**	**Pop**	**Hotp**	**2Wk**	**MuS**	**Bam**	**Avg**
Vanilla	24.6	48.1	31.0	26.3	25.3	7.2	59.7	31.7	29.4	46.9	37.0	23.5	19.6	6.4	10.5	24.8
GRPO	39.3	60.6	41.1	37.4	34.6	15.4	26.4	36.4	40.0	58.9	43.5	35.4	30.3	12.0	65.7	40.8
G-OPSD	**44.9**	61.2	45.2	40.4	38.5	16.0	66.1	44.6	40.7	58.9	45.0	37.0	34.6	13.3	65.7	42.2
Skill-SD	44.4	60.4	44.0	39.5	40.4	15.4	64.9	44.1	39.1	57.5	45.4	34.8	34.1	10.7	64.1	40.8
RLSD	41.5	58.6	42.3	40.4	40.2	**16.8**	**66.9**	43.8	38.6	57.3	43.0	34.5	34.1	11.5	65.3	40.6
SDAR	44.8	58.1	44.3	38.6	36.2	15.7	66.1	43.4	39.7	58.9	45.3	35.9	35.5	12.6	65.3	41.9
PCSD	40.5	59.8	42.7	38.4	39.3	15.3	63.7	42.8	41.9	59.2	43.9	36.5	36.0	11.2	63.7	41.8
PESD	44.8	**62.3**	**46.6**	**41.9**	**43.5**	16.3	**66.9**	**46.0**	**44.7**	**60.2**	**47.4**	**37.9**	**38.2**	**13.5**	**66.1**	**44.0**



(a) ALFWorld k^* (b) ALFWorld $\mathcal{P}/\mathcal{N}$ (c) Search-QA k^* (d) Search-QA $\mathcal{P}/\mathcal{N}$

Figure 3: Dynamics of privileged supervision. (a,c) Mean privilege level k^* per training batch. (b,d) Fractions of failed actions assigned to $\mathcal{P}$ (solid) and $\mathcal{N}$ (dashed); the remaining actions fall in $\mathcal{U}$. Translucent traces show batch statistics and opaque curves show moving averages.

5 Ablations and Diagnostics

We ablate PESD on ALFWorld with Qwen2.5-3B-Instruct to examine the contributions of failure supervision, privileged experience, and teacher construction. Beyond standard component ablations, we provide diagnostics on the $\mathcal{P}/\mathcal{N}/\mathcal{U}$ partition and on the consistency threshold that controls reverse supervision.

5.1 Component Ablations

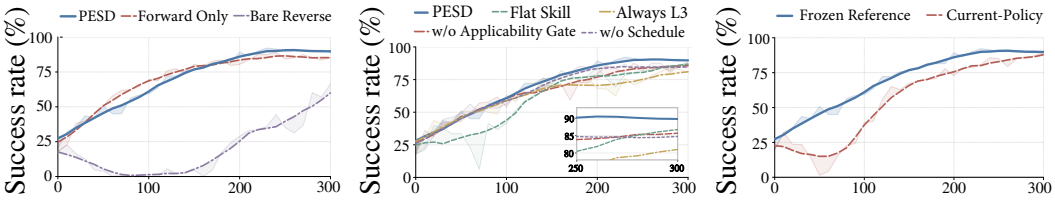
For failure supervision, *w/o Reverse* removes unlikelihood on $\mathcal{N}$, while *Bare Reverse* enables permissive reverse supervision from the start by removing warmup and admission controls. For privileged experience, *w/o Applicability Gate* always admits the retrieved principle, *Flat Skill* replaces the cumulative hierarchy with a single gated skill, *Always* L_3 fixes the strongest available level, and *w/o Progressive Schedule* keeps L_k unchanged across revisits. *Current-Policy Teacher* replaces only the frozen teacher on successful trajectories; the failure branch remains unchanged.

Table 3: Ablations on ALFWorld.

Variant	**Average**
PESD	**92.2**
Failure supervision	
w/o Reverse	89.1
Bare Reverse	66.4
Privileged experience	
w/o Applicability Gate	87.5
Flat Skill	88.3
Always L_3	84.4
w/o Progressive Schedule	87.5
Teacher construction	
Current-Policy Teacher	86.7

Failure supervision. Removing reverse supervision lowers the average from 92.2 to 89.1. Appendix D.3 further separates the two failure-side branches: removing all failure-side supervision also yields 89.1, reverse supervision alone reaches 90.6, and combining forward and reverse supervision achieves 92.2. Forward supervision therefore provides little benefit on its own in this run, but improves performance when coupled with the reverse branch. Since each variant changes the policy and thus the subsequent on-policy rollouts and partition assignments, the effects of the two branches need not be additive. In contrast, *Bare Reverse* drops to 66.4 and is unstable early in training (Figure 4a), showing that negative supervision is most effective when introduced selectively and conservatively.

Privileged context. Removing the applicability gate, flattening the privilege hierarchy, or keeping the guidance fixed reduces the average from 92.2 to 84.4–88.3. As shown in Figure 4b, these variants often improve early but converge below the full method. The



(a) Failure supervision (b) Privileged context (c) Teacher construction

Figure 4: Training dynamics of the ablation variants on ALFWorld with Qwen2.5-3B-Instruct.

results favor combining applicability-filtered experience with a privilege level that changes across repeated attempts, rather than exposing the policy to a fixed form of guidance throughout training.

Teacher construction. Figure 4c isolates the teacher used on successful trajectories. Replacing the frozen reference checkpoint with the current policy causes a pronounced early drop and reaches a lower peak. We therefore retain the frozen teacher for consolidating successful behavior and use the rollout policy only for failure-aware scoring.

5.2 Action Partition Validation

The component ablations show that reverse supervision is useful when applied selectively, but do not reveal what kinds of actions enter $\mathcal{N}$. We therefore manually annotate 20 failed ALFWorld episodes, labeling each action as correct, wrong, invalid, or unclear independently of its PESD assignment. Under the default setting, most actions remain in $\mathcal{U}$ and only a small subset enters $\mathcal{N}$, with limited overlap between $\mathcal{N}$ and verified useful progress. This supports interpreting $\mathcal{P}/\mathcal{N}/\mathcal{U}$ as supervision regions rather than semantic action classes. Full statistics are provided in Appendix D.

We further vary the consistency threshold ρ, which controls how readily actions are admitted into $\mathcal{N}$. Lowering ρ from 0.6 to 0.3 broadens reverse supervision but still reaches 91.4, outperforming the 89.1 of *w/o Reverse* and confirming that negative supervision remains beneficial under a moderately relaxed gate. In contrast, setting $\rho = 0$ reduces performance to 85.9. Without a consistency constraint, more useful actions are admitted into $\mathcal{N}$ and incorrectly suppressed, offsetting the gains from reverse supervision. These results indicate that negative supervision is beneficial when reverse updates are restricted to actions supported by sufficiently consistent negative evidence.

Table 4: Effect of ρ on ALFWorld.

ρ	0.60	0.30	0.00
Avg.	92.2	91.4	85.9

6 Limitations.

PESD derives action-level supervision from privileged hindsight, but the resulting $\mathcal{P}/\mathcal{N}/\mathcal{U}$ partition captures context-induced likelihood shifts rather than causal credit assignment. Its quality therefore depends on whether the privileged context provides reliable evidence about the realized behavior. The conservative partition and consistency gate limit the use of ambiguous negative signals, but noisy retrieved experience or imperfect reference trajectories may still distort the supervision. Future work could study how to identify unreliable privileged evidence and adapt the privilege schedule when retrieved skills or reference trajectories are noisy, incomplete, or unavailable.

7 Conclusion

We introduced PESD, a failure-aware approach for learning from mixed-quality on-policy trajectories. By combining progressive privileged guidance with selective forward, reverse, and RL-only supervision, PESD extracts useful training signal from failed rollouts without treating every action uniformly. Experiments on ALFWorld and Search-QA show strong performance across two model scales, while the ablations support the roles of conservative failure supervision, applicability filtering, and progressive privilege reduction. More broadly, our results suggest that failed trajectories need not be treated as globally negative: useful behavior and corrective evidence can coexist within the same rollout and can be separated through privileged re-scoring.

AI USE STATEMENT

We used AI tools to assist with writing and language polishing, including improving clarity, grammar, and presentation. We also used them for literature retrieval and discovery, such as identifying potentially relevant prior work, and for auxiliary tasks including code review, formatting checks, and proofreading. Generative AI was not used to generate experimental results or make final scientific judgments. We take full responsibility for the content of this paper, including any text or artifacts produced with the aid of AI tools.

ETHICS STATEMENT

This work studies training methods for LLM agents in simulated household and information-retrieval environments. It does not involve human subjects, private user data, or deployment in safety-critical settings. The experiments use existing benchmarks and models for research purposes.

REPRODUCIBILITY STATEMENT

We provide the main algorithm, training objective, and model design in the main paper, together with implementation details, hyperparameters, retrieval and applicability-gating procedures, and additional analyses in the appendix. The experimental setup specifies the evaluated benchmarks, model backbones, training configurations, and hardware. Additional ablations and diagnostic results are included to clarify the contribution of individual components. We will release the code and implementation materials to facilitate reproduction of the reported results.

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.

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

Jose A Arjona-Medina, Michael Gillhofer, Michael Widrich, Thomas Unterthiner, Johannes Brandstetter, and Sepp Hochreiter. Rudder: Return decomposition for delayed rewards. *Advances in Neural Information Processing Systems*, 32, 2019.

Ken Ding. Hdpo: Hybrid distillation policy optimization via privileged self-distillation, 2026. URL https://arxiv.org/abs/2603.23871.

Guanting Dong, Hangyu Mao, Kai Ma, Licheng Bao, Yifei Chen, Zhongyuan Wang, Zhongxia Chen, Jiazhen Du, Huiyang Wang, Fuzheng Zhang, et al. Agentic reinforced policy optimization. In *International Conference on Learning Representations*, volume 2026, pp. 16981–17017, 2026.

Lang Feng, Zhenghai Xue, Tingcong Liu, and Bo An. Group-in-group policy optimization for llm agent training. *Advances in Neural Information Processing Systems*, 38:46375–46408, 2026.

Yubin Ge, Salvatore Romeo, Jason Cai, Monica Sunkara, and Yi Zhang. Samule: Self-learning agents enhanced by multi-level reflection. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 16602–16621, 2025.

Jonas Gehring, Kunhao Zheng, Jade Copet, Vegard Mella, Quentin Carbonneaux, Taco Cohen, and Gabriel Synnaeve. Rlef: Grounding code llms in execution feedback with reinforcement learning. *arXiv preprint arXiv:2410.02089*, 2024.

Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. Minillm: Knowledge distillation of large language models. In *International Conference on Learning Representations*, volume 2024, pp. 32694–32717, 2024.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

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

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. Constructing a multi-hop qa dataset for comprehensive evaluation of reasoning steps. In *Proceedings of the 28th International Conference on Computational Linguistics*, pp. 6609–6625, 2020.

Dongwei Jiang, Jingyu Zhang, Orion Weller, Nathaniel Weir, Benjamin Van Durme, and Daniel Khashabi. Self-[in]correct: Llms struggle with discriminating self-generated responses, 2024. URL https://arxiv.org/abs/2404.04298.

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.

Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1601–1611, 2017.

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

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466, 2019.

Yaxuan Li, Yuxin Zuo, Bingxiang He, Jinqian Zhang, Chaojun Xiao, Cheng Qian, Tianyu Yu, Huan ang Gao, Wenkai Yang, Zhiyuan Liu, and Ning Ding. Rethinking on-policy distillation of large language models: Phenomenology, mechanism, and recipe, 2026a. URL https://arxiv.org/abs/2604.13016.

Zongxia Li, Zhongzhi Li, Yucheng Shi, Ruhan Wang, Junyao Yang, Zhichao Liu, Xiyang Wu, Anhao Li, Yue Yu, Ninghao Liu, et al. Long-horizon-terminal-bench: Testing the limits of agents on long-horizon terminal tasks with dense reward-based grading. *arXiv preprint arXiv:2607.08964*, 2026b.

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

Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, et al. Agentbench: Evaluating llms as agents. In *International Conference on Learning Representations*, volume 2024, pp. 52989–53046, 2024.

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, 2026a. URL https://arxiv.org/abs/2605.15155.

Zhengxi Lu, Zhiyuan Yao, Jinyang Wu, Chengcheng Han, Qi Gu, Xunliang Cai, Weiming Lu, Jun Xiao, Yueting Zhuang, and Yongliang Shen. Skill0: In-context agentic reinforcement learning for skill internalization, 2026b. URL https://arxiv.org/abs/2604.02268.

Junyu Luo, Weizhi Zhang, Ye Yuan, Yusheng Zhao, Junwei Yang, Yiyang Gu, Bohan Wu, Binqi Chen, Ziyue Qiao, Qingqing Long, et al. Large language model agent: A survey on methodology, applications and challenges. *arXiv preprint arXiv:2503.21460*, 2025a.

Xufang Luo, Yuge Zhang, Zhiyuan He, Zilong Wang, Siyun Zhao, Dongsheng Li, Luna K Qiu, and Yuqing Yang. Agent lightning: Train any ai agents with reinforcement learning. *arXiv preprint arXiv:2508.03680*, 2025b.

Chunji Lv, Yangguang Wei, Junlin Liu, Yang Gao, Ming Liu, Xinming Wang, Jinyang Wu, Guoren Wang, and Changsheng Li. Pcsd: Persistent consistency for self-distillation in agentic reinforcement learning, 2026. URL https://arxiv.org/abs/2608.01837.

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

Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. In *Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: Long papers)*, pp. 9802–9822, 2023.

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

Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A Smith, and Mike Lewis. Measuring and narrowing the compositionality gap in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 5687–5711, 2023.

Zehan Qi, Xiao Liu, Iat Long Iong, Hanyu Lai, Xueqiao Sun, Jiadai Sun, Xinyue Yang, Yu Yang, Shuntian Yao, Wei Xu, et al. Webrl: Training llm web agents via self-evolving online curriculum reinforcement learning. In *International Conference on Learning Representations*, volume 2025, pp. 79791–79821, 2025.

Cheng Qian, Emre Can Acikgoz, Qi He, Hongru Wang, Xiusi Chen, Dilek Hakkani-Tur, Gokhan Tur, and Heng Ji. Toolrl: Reward is all tool learning needs. *Advances in Neural Information Processing Systems*, 38:105523–105553, 2026.

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

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

Zhengliang Shi, Shen Gao, Lingyong Yan, Yue Feng, Xiuyi Chen, Zhumin Chen, Dawei Yin, Suzan Verberne, and Zhaochun Ren. Tool learning in the wild: Empowering language models as automatic tool agents. In *Proceedings of the ACM on Web Conference 2025*, pp. 2222–2237, 2025.

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

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. Alfworld: Aligning text and embodied environments for interactive learning. *arXiv preprint arXiv:2010.03768*, 2020.

Xiangru Tang, Tianrui Qin, Tianhao Peng, Ziyang Zhou, Daniel Shao, Tingting Du, Xinming Wei, He Zhu, Ge Zhang, Jiaheng Liu, Xingyao Wang, Sirui Hong, Chenglin Wu, and Wangchunshu Zhou. AGENT KB: A hierarchical memory framework for cross-domain agentic problem solving. In *ICML 2025 Workshop on Collaborative and Federated Agentic Workflows*, 2025. URL https://openreview.net/forum?id=ohXoWHlrn8.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. Musique: Multihop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554, 2022.

Jonathan Uesato, Nate Kushman, Ramana Kumar, Francis Song, Noah Siegel, Lisa Wang, Antonia Creswell, Geoffrey Irving, and Irina Higgins. Solving math word problems with process-and outcome-based feedback. *arXiv preprint arXiv:2211.14275*, 2022.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models. *arXiv preprint arXiv:2305.16291*, 2023.

Hao Wang, Guozhi Wang, Han Xiao, Yufeng Zhou, Yue Pan, Jichao Wang, Ke Xu, Yafei Wen, Xiaohu Ruan, Xiaoxin Chen, and Honggang Qi. Skill-sd: Skill-conditioned self-distillation for multi-turn llm agents, 2026a. URL https://arxiv.org/abs/2604.10674.

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*, 2026b. URL https://openreview.net/forum?id=KH9eJRz8WR.

Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. Text embeddings by weakly-supervised contrastive pre-training. *arXiv preprint arXiv:2212.03533*, 2022.

Sinuo Wang, Wang PIAOHONG, Tianrui Qin, Maojia Song, Qianben Chen, Qiexiang Wang, Gengze Zhou, Zeyu Zhang, He Zhu, Dingfeng Shi, Yutong Xie, Minghao Liu, Jiaheng Liu, Ge Zhang, Jiawei Ma, Yuchen Eleanor Jiang, Qi Wu, and Wangchunshu Zhou. EVOLVING ROLLOUTS: Harnessing historical experience for web agent evolution in reinforcement learning. In *Forty-third International Conference on Machine Learning*, 2026c. URL https://openreview.net/forum?id=p5B4ABjuT2.

Zihan Wang, Kangrui Wang, Qineng Wang, Pingyue Zhang, Linjie Li, Zhengyuan Yang, Xing Jin, Kefan Yu, Minh Nhat Nguyen, Licheng Liu, et al. Ragen: Understanding self-evolution in llm agents via multi-turn reinforcement learning. *arXiv preprint arXiv:2504.20073*, 2025.

Sean Welleck, Ilia Kulikov, Stephen Roller, Emily Dinan, Kyunghyun Cho, and Jason Weston. Neural text generation with unlikelihood training. *arXiv preprint arXiv:1908.04319*, 2019.

Jinyang Wu, Shuo Yang, Zhengxi Lu, Fan Zhang, Yuhao Shen, Lang Feng, Haoran Luo, Zheng Lian, Shuai Zhang, Zhengqi Wen, and Jianhua Tao. Seed: Self-evolving on-policy distillation for agentic reinforcement learning, 2026a. URL https://arxiv.org/abs/2607.14777.

Jinyang Wu, Shuo Yang, Yuhao Shen, Shuai Zhang, Zhengqi Wen, and Jianhua Tao. Spark: Strategic policy-aware exploration via dynamic branching for long-horizon agentic learning. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 23981–24004, 2026b.

Zhiheng Xi, Wenxiang Chen, Xin Guo, Wei He, Yiwen Ding, Boyang Hong, Ming Zhang, Junzhe Wang, Senjie Jin, Enyu Zhou, et al. The rise and potential of large language model based agents: A survey. *Science China information sciences*, 68(2):121101, 2025.

Peng Xia, Jianwen Chen, Hanyang Wang, Jiaqi Liu, Kaide Zeng, Yu Wang, Siwei Han, Yiyang Zhou, Xujiang Zhao, Haifeng Chen, Zeyu Zheng, Cihang Xie, and Huaxiu Yao. Skillrl: Evolving agents via recursive skill-augmented reinforcement learning, 2026. URL https://arxiv.org/abs/2602.08234.

Hang Yan, Fangzhi Xu, Qiushi Sun, Jinyang Wu, Zixian Huang, Muye Huang, Jingyang Gong, Zichen Ding, Kanzhi Cheng, Yian Wang, et al. Odysseyarena: Benchmarking large language models for long-horizon, active and inductive interactions. *arXiv preprint arXiv:2602.05843*, 2026.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang

Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025a. URL https://arxiv.org/abs/2505.09388.

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

Chenxu Yang, Chuanyu Qin, Qingyi Si, Minghui Chen, Naibin Gu, Dingyu Yao, Zheng Lin, Weiping Wang, Jiaqi Wang, and Nan Duan. Self-distilled rlvr, 2026a. URL https://arxiv.org/abs/2604.03128.

Shuo Yang, Jinyang Wu, Zhengxi Lu, Yuhao Shen, Fan Zhang, Lang Feng, Shuai Zhang, Haoran Luo, Zheng Lian, Zhengqi Wen, and Jianhua Tao. Opid: On-policy skill distillation for agentic reinforcement learning, 2026b. URL https://arxiv.org/abs/2606.26790.

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*, 2026c.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D Manning. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pp. 2369–2380, 2018.

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

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, et al. Dapo: An open-source llm reinforcement learning system at scale. *Advances in Neural Information Processing Systems*, 38:113222–113244, 2026.

Yinger Zhang, Shutong Jiang, Renhao Li, Jianhong Tu, Yang Su, Lianghao Deng, Xudong Guo, Chenxu Lv, and Junyang Lin. Deepplanning: Benchmarking long-horizon agentic planning with verifiable constraints. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 7377–7407, 2026.

Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. Expel: Llm agents are experiential learners. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2024.

Siyan Zhao, Zhihui Xie, Mengchen Liu, Jing Huang, Guan Pang, Feiyu Chen, and Aditya Grover. Self-distilled reasoner: On-policy self-distillation for large language models. In *ICLR 2026 Workshop on Lifelong Agents: Learning, Aligning, Evolving*, 2026. URL https://openreview.net/forum?id=31wkVZXEaJ.

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, 2026. URL https://arxiv.org/abs/2605.07725.

A Alternative Sample-Level Aggregation

The main Search-QA results follow the evaluation pipeline used by SDAR and subsequent work, which reports the average over evaluation slices. To complement this protocol, we additionally report sample-level aggregation, where each question contributes equally to the final score.

The two aggregation schemes are nearly identical when evaluation slices have similar sizes, but can differ when slice sizes are highly uneven. This effect is most visible on Bamboogle, whose 125 examples are divided into one slice of 124 examples and one singleton slice. If k of the first 124 examples are correct and the final example has reward $r_{125} \in \{0, 1\}$, the slice-level score is

$$\frac{1}{2}\left(\frac{k}{124} + r_{125}\right), \tag{9}$$

while sample-level aggregation gives

$$\frac{k + r_{125}}{125}. \tag{10}$$

We retain the established slice-level protocol in the main results for consistency with prior work and report sample-level aggregation here.

Table 5: Reported and sample-level performance on Bamboogle.

	Qwen2.5-3B-Instruct		Qwen3-1.7B	
Method	**Reported**	**Sample-level**	**Reported**	**Sample-level**
Vanilla	59.7	20.0 (-39.7)	10.5	21.0 (+10.5)
G-OPSD	66.1	32.8 (-33.3)	65.7	32.0 (-33.7)
Skill-SD	64.9	30.4 (-34.5)	64.1	28.8 (-35.3)
RLSD	66.9	34.4 (-32.5)	65.3	31.2 (-34.1)
SDAR	66.1	32.8 (-33.3)	65.3	31.2 (-34.1)
PCSD	63.7	27.4 (-36.3)	63.7	27.4 (-36.3)
PESD	66.9	34.4 (-32.5)	66.1	32.8 (-33.3)

Table 6 further reports the full sample-level Search-QA results for PESD. The largest numerical difference appears on Bamboogle, while the remaining datasets are comparatively stable across the two aggregation schemes.

Table 6: Reported and sample-level Search-QA results for PESD. Parenthesized values show the change from the reported score.

Backbone	Evaluation	NQ	Triv	Pop	Hotp	2Wk	MuS	Bam	Avg
3B	Reported	44.8	62.3	46.6	41.9	43.5	16.3	66.9	46.0
	Sample-level	46.1 (+1.3)	62.5 (+0.2)	44.7 (-1.9)	42.0 (+0.1)	43.3 (-0.2)	16.5 (+0.2)	34.4 (-32.5)	41.4 (-4.6)
1.7B	Reported	44.7	60.2	47.4	37.9	38.2	13.5	66.1	44.0
	Sample-level	44.5 (-0.2)	60.4 (+0.2)	46.1 (-1.3)	38.0 (+0.1)	38.0 (-0.2)	13.8 (+0.3)	32.8 (-33.3)	39.1 (-4.9)

The sample-level results provide an additional view of PESD under question-level weighting. Since complete sample-level values are not available for every baseline–backbone pair, we report them as a complementary aggregation rather than use them for cross-method ranking.

B Case Study

We illustrate the teacher inputs used by SDAR and PESD on the same failed ALFWorld trajectory. The task is `cool some lettuce and put it in countertop`. The rollout makes partial progress but reaches placement without completing the required cooling operation.

Under skill-conditioned SDAR (Lu et al., 2026a), the retrieved skill is *Systematic Object Hunt*, which encourages search over unvisited surfaces and containers but does not specify the omitted cooling operation. PESD first passes this skill through the applicability gate, which admits the original principle to L_2 only when it is sufficiently relevant to the current trajectory. The teacher also receives the completed failed episode as c^{traj}; on this first failure visit, L_3 contains a verified successful trajectory in which cooling precedes placement. Instance-specific contrast therefore comes

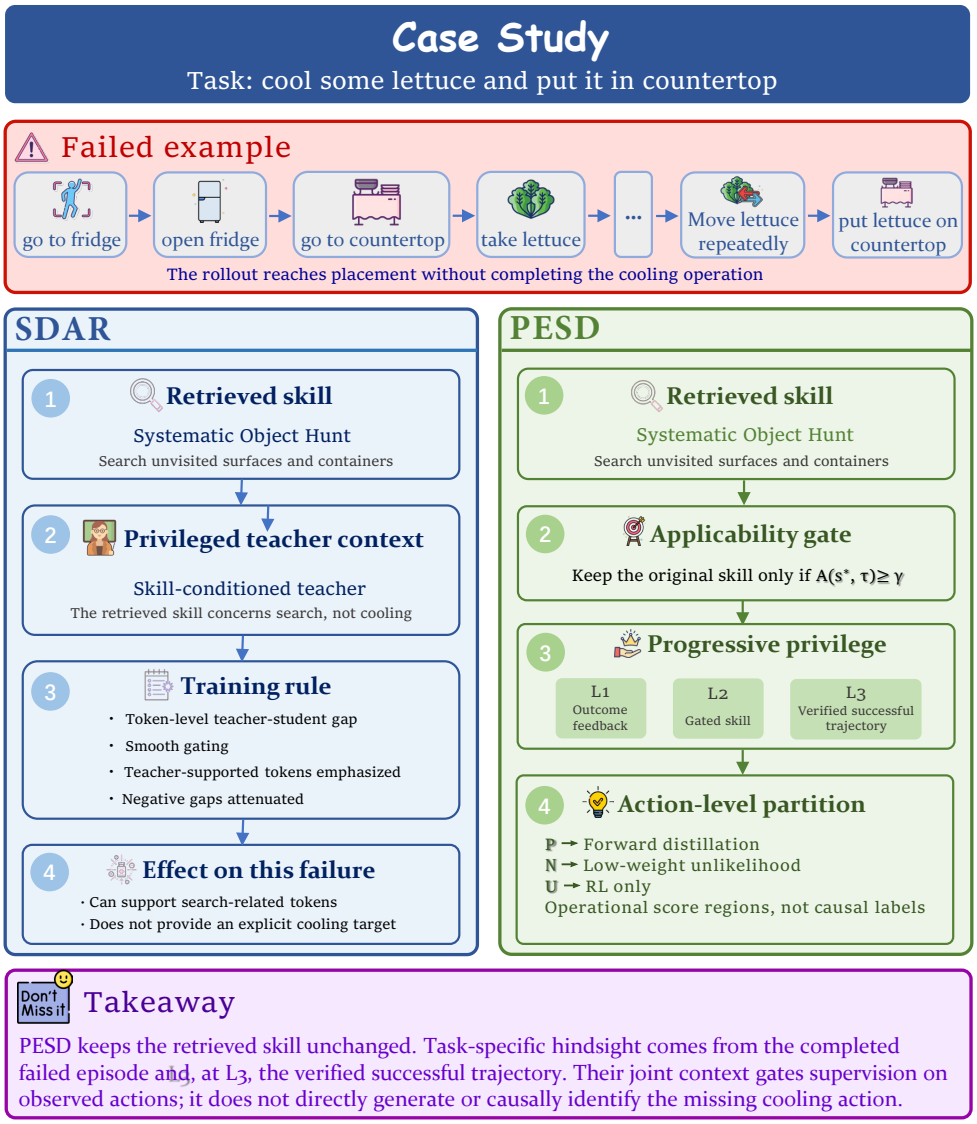


Figure 5: Teacher-side information flow for a failed ALFWorld rollout. PESD leaves the retrieved skill unchanged and supplies instance-specific hindsight through the completed episode and, at L_3, a verified successful trajectory.

from c^{traj} and the verified reference rather than a rewritten skill. Their combined effect re-scores the observed action spans and produces the $\mathcal{P}/\mathcal{N}/\mathcal{U}$ partition.

C Method Details

This section specifies the mechanics of PESD's failure-aware on-policy distillation. Successful behavior is scored by a frozen teacher, whereas failed behavior is re-scored under the rollout policy that generated it. The resulting likelihood gap assigns each observed action to forward distillation, reverse unlikelihood, or the RL objective alone. Retrieved skills enter only after an applicability gate, and the full teacher context combines the completed episode c^{traj} with the scheduled ladder L_k. As in Section 3.4, the partition describes context-induced score changes rather than causal action quality. We implement the policy update with on-policy GRPO and use existing gated-distillation, persistence-weighting, and SkillBank components (Lu et al., 2026a; Lv et al., 2026; Xia et al., 2026) as infrastructure. These components provide convenient scoring and memory mechanisms;

the supervision rule, in particular the signed treatment of failed actions, is the PESD objective described in Section 3.

C.1 Teacher Construction and Gradient Flow

We use different teacher constructions for failed and successful trajectories. In both cases, the distillation target is the student's own on-policy trajectory; privileged information changes how these tokens are scored but never replaces them with teacher-generated actions.

Failure trajectories. For each training batch, let θ_{old} denote the policy parameters used to generate the on-policy trajectories. On failed trajectories, the privileged and unprivileged scores are computed using the same policy snapshot. Specifically, we define $\bar{\theta} = \text{sg}(\theta_{\text{old}})$, where $\text{sg}(\cdot)$ denotes stop-gradient, and compute

$$r_{t,j}^{\text{fail}} = \text{sg}[\log \pi_{\bar{\theta}}(y_{t,j} \mid h_{t,j}, C_k) - \log \pi_{\bar{\theta}}(y_{t,j} \mid h_{t,j})] \,. \tag{11}$$

Thus, the score measures how the same rollout snapshot changes its likelihood for the realized token when conditioned on the full privileged context $C_k = (c^{\text{traj}}, L_k)$.

Using the same rollout policy in the privileged and unprivileged branches isolates the effect of the hindsight context. The unprivileged branch conditions only on the original causal history $h_{t,j}$, whereas the privileged branch additionally receives the completed failed episode, including the realized action, subsequent actions, and the final outcome, together with the scheduled guidance in L_k. When L_3 is active, a verified successful trajectory is included as additional context. Their likelihood difference measures how hindsight changes the model's assessment of the realized behavior, rather than the causal contribution of any individual action.

Successful trajectories. On successes, PESD keeps a frozen privileged teacher so that the support signal does not drift with the improving policy. Let θ_{ref} denote the frozen reference checkpoint initialized before RL training. The teacher receives C_k, while the current policy remains the student. For an observed successful token, we compute the detached support signal

$$r_{t,j}^{\text{succ}} = \text{sg}[\log \pi_{\theta_{\text{ref}}}(y_{t,j} \mid h_{t,j}, C_k) - \log \pi_{\theta}(y_{t,j} \mid h_{t,j})] \,. \tag{12}$$

Unlike Equation (11), the two terms use different models: a frozen privileged teacher versus the current unprivileged student. The reference model is frozen throughout training, and the score is used only to construct token weights. The student log-likelihood in the optimization objective remains differentiable.

Successful trajectories are not partitioned by the action-level thresholds τ_+ and τ_-. Token weights come from the PCSD persistence gate applied to Equation (12), with the environment-facing action span as evidence (Appendix C.4). Isolated teacher-support spikes therefore do not dominate the update. Successful trajectories never enter the reverse branch. The two constructions are summarized in Table 7.

Table 7: Teacher construction for successful and failed on-policy trajectories. In both cases, the target tokens are generated by the student policy.

Trajectory	Scoring teacher	Unprivileged term	Distillation
Success	frozen $\pi_{\theta_{\text{ref}}}(\cdot \mid h, C_k)$	current $\pi_\theta(\cdot \mid h)$	PCSD forward only
Failure	$\text{sg}(\pi_{\theta_{\text{old}}}(\cdot \mid h, C_k))$	$\text{sg}(\pi_{\theta_{\text{old}}}(\cdot \mid h))$	forward / reverse / none

C.2 Skill Retrieval and Teacher Context

Retrieval. Reusable skills are stored in a SkillBank (Xia et al., 2026). Each skill contains a reusable principle together with information about its applicability and the trajectories from which it was derived. PESD does not retrieve from the task alone: the query includes the current rollout, so that different failures of the same task can select different experience. For failed trajectories, the query additionally emphasizes the observed failure state or unresolved decision. We retain the highest-ranked skill as s^*.

Retrieval provides a reusable prior. Whether that prior enters L_2 is decided by the applicability gate below, not by rewriting the skill into an actor prompt.

Applicability gate. We use a lightweight lexical gate to determine whether a retrieved skill should enter L_2. Let $\mathcal{T}(s^*)$ denote the set of unique alphanumeric tokens of length at least four extracted from the skill name, applicability, principle, and procedure. We construct $\sigma(\tau)$ from the non-empty step strings in the current trajectory, retaining all steps when there are at most eight and otherwise keeping the first four and last four. The applicability score is

$$A(s^*, \tau) = \min\left(1,\ \frac{|\{t \in \mathcal{T}(s^*) : t \text{ occurs in } \sigma(\tau)\}|}{|\mathcal{T}(s^*)|}\right), \tag{13}$$

with $A(s^*, \tau) = 0$ when $\mathcal{T}(s^*)$ is empty.

With $\gamma = 0.35$, we set

$$e^{\text{skill}} = \begin{cases} \text{principle}(s^*), & A(s^*, \tau) \geq \gamma \text{ and } \text{principle}(s^*) \neq \varnothing, \\ \varnothing, & \text{otherwise.} \end{cases} \tag{14}$$

The gate leaves the retrieved skill unchanged and only controls whether its principle is included in the privileged context. In the *w/o Applicability Gate* ablation, we set $\gamma = 0$, so every non-empty retrieved principle is included in L_2.

Episode context. The scorer conditions on $C_k = (c^{\text{traj}}, L_k)$, which combines completed-episode hindsight with the scheduled privilege level. For a failed trajectory, c^{traj} is the student's completed failed episode, while L_k supplies additional outcome-, skill-, or reference-level guidance. The full episode is used intentionally as hindsight: when evaluating a_t, it includes the realized action itself and may also include subsequent actions and the unsuccessful outcome, while the acting history $h_{t,j}$ remains unchanged and causal. For a successful trajectory, c^{traj} is a verified success from the same GRPO group (shortest by default), likewise provided as a completed episode.

The ladder is then appended to the episode context: L_1 provides outcome feedback, L_2 adds the gated skill, and L_3 further includes a textual rendering of τ^{ref}. A first-visit success therefore already receives a working episode through c^{traj}; L_3 adds an additional verified trajectory only when the schedule, or the *Always* L_3 variant, selects $k = 3$. The *Always* L_3 ablation changes only the ladder, not the construction of c^{traj}. Because c^{traj} already provides instance-specific hindsight, retaining τ^{ref} at every visit need not be uniformly beneficial; this is precisely what the *Always* L_3 ablation tests.

The teacher context is rebuilt for every newly generated trajectory. When a task is revisited after the policy has changed, retrieval, gating, and C_k are recomputed from the new rollout.

C.3 Failure Action Partition

The three-way partition in Equation (5) is applied *only* to failed trajectories. We aggregate the detached privilege gap from Equation (11) over the environment-facing action span $\mathcal{A}_t$ (the ⟨action⟩ / ⟨search⟩ / ⟨answer⟩ block, or otherwise tokens after the last ⟨/think⟩),

$$R_t = \frac{1}{|\mathcal{A}_t|} \sum_{j \in \mathcal{A}_t} r^{\text{fail}}_{t,j}. \tag{15}$$

An action enters the positive branch when $R_t > \tau_+$ and the reverse branch when $R_t < -\tau_-$ with persistent negative evidence. We define the set of tokens with nonzero likelihood shifts as

$$\mathcal{S}_t = \left\{ j \in \mathcal{A}_t : |r^{\text{fail}}_{t,j}| > 0 \right\}, \tag{16}$$

and measure the consistency of negative evidence by

$$c_t^- = \frac{1}{|\mathcal{S}_t|} \sum_{j \in \mathcal{S}_t} \mathbb{I}[r^{\text{fail}}_{t,j} < 0], \tag{17}$$

with $c_t^- = 0$ when $\mathcal{S}_t$ is empty. Since reverse admission already requires $R_t < -\tau_-$, c_t^- is used only to measure sign agreement among tokens affected by the privileged context. We require $c_t^- \geq \rho$ for reverse supervision. Actions in the deadband $\mathcal{U}$ receive no distillation update and are optimized only through RL.

We intentionally make the negative criterion stricter than the positive one. In our default configuration, $\tau_+ = 0.05$, $\tau_- = 0.20$, and $\rho = 0.60$. The larger magnitude required for negative evidence, with the consistency requirement, reduces the chance of suppressing an action based on weak or conflicting privileged signals.

Successful trajectories do *not* use this partition. Their token weights come from the PCSD persistence gate described next.

C.4 Distillation Weights and Optimization

Forward path. PESD first determines which actions are eligible for positive supervision, and then applies persistence-weighted distillation (Lv et al., 2026). Teacher-support signals are aggregated over an adaptive local window, with a one-sided trend factor that discounts rapidly collapsing likelihood shifts, so isolated positive spikes do not dominate the update. Aggregation is restricted to the environment-facing action span, and the resulting weight w^+ is detached from optimization. With a stop-gradient teacher, the gated reverse-KL objective (Lu et al., 2026a) reduces to the weighted on-policy NLL used in Equation (6).

The same positive weighting is used in two settings:

- **Success.** All on-policy tokens in a successful trajectory are eligible for PCSD weighting; no $R_t > \tau_+$ action filter is applied.
- **Fail-forward.** Only action spans satisfying $R_t > \tau_+$ are eligible, after which PCSD weighting is applied within those spans.

An optional hierarchy-residual factor can further attenuate tokens for which moving from L_{k-1} to L_k weakens teacher support. We omit this extra teacher pass in the default signed-failure setting.

Reverse path. Reverse supervision is applied only to failed actions assigned to $\mathcal{N}$. The trigger is defined at the action level, while suppression is applied selectively at the token level. For an action at step t, we define

$$w_t^- = \mathbf{1}[R_t < -\tau_-]\mathbf{1}[c_t^- \geq \rho], \tag{18}$$

where $\tau_- = 0.20$ and $\rho = 0.60$. Thus, w_t^- is a binary action-level trigger rather than a continuous weight.

For each triggered action, token weights are determined by the magnitude of the negative privileged shift:

$$w_{t,j}^- = w_t^- \cdot \frac{[-r_{t,j}^{\text{fail}}]_+}{\text{mean}_{u \in \mathcal{A}_t:\, r_{t,u}^{\text{fail}} < 0}[-r_{t,u}^{\text{fail}}]_+}, \tag{19}$$

where $\mathcal{A}_t$ denotes the environment-facing token span of action a_t.

For tokens with nonzero reverse weight, we apply the unlikelihood objective of Welleck et al. (2019),

$$-\log(1 - \pi_\theta(y_{t,j} \mid h_{t,j}))\,.$$

This directly lowers the probability of the observed token without requiring an alternative target, while RL continues to optimize the trajectory-level objective.

Coefficients and schedule. In Equations (7) and (8), we set $\lambda_{\text{PESD}} = 0.01$ and $\lambda_{\text{rev}} = 0.1$, yielding effective coefficients of 0.01 for forward distillation and 0.001 for reverse supervision.

Failure distillation is disabled for the first 30 optimization steps and linearly warmed up over the next 20. Its strength is further modulated by an exponential moving average of the training-batch success rate. Let s_b denote the success rate of the current batch. We initialize $\bar{s}$ with the first observed value and subsequently update it as

$$\bar{s} \leftarrow 0.9\,\bar{s} + 0.1\,s_b.$$

The failure branch is disabled when $\bar{s} \leq 0.03$, fully enabled when $\bar{s} \geq 0.10$, and linearly interpolated between the two thresholds. Together with the warmup schedule, this defines α_u in Equation (8). Success distillation remains active throughout training.

Bare Reverse. The *Bare Reverse* ablation removes these stabilization mechanisms by setting `warmup=ramp=0`, `fuse_high=0`, and `support_min=0`. Reverse supervision is therefore active from

the beginning under permissive admission settings. We use this variant as a stress test of early, weakly filtered reversal rather than as an isolated component ablation.

All RL trajectories remain on-policy. Teacher scores, retrieved experience, and privilege states are auxiliary quantities used only to construct the distillation objective. For failed trajectories, the stop-gradient scorer $\pi_{\bar{\theta}}$ is recreated from the policy that generated the current batch rather than maintained as a separately trained teacher.

C.5 Reference Trajectory Selection

When multiple verified successful rollouts are available for the same task, we use the shortest trajectory as τ^{ref}. Once success has been verified by the environment, a shorter rollout typically provides a more direct solution with fewer exploratory or redundant actions. It also reduces the amount of privileged context required for teacher-side scoring.

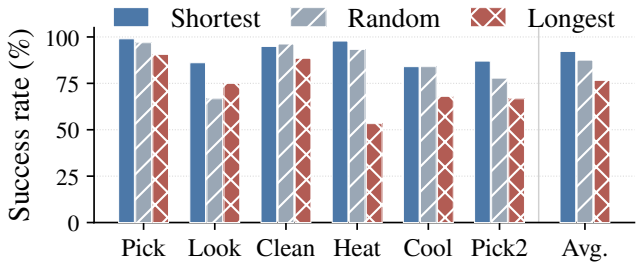


Figure 6: Reference trajectory selection on ALFWorld with Qwen2.5-3B.

We compare this choice with random and longest-reference selection on ALFWorld using Qwen2.5-3B. As shown in Figure 6, the shortest successful rollout performs best while also providing the most compact reference. Longer trajectories tend to contain exploratory or redundant actions that are not necessary for task completion, while random selection may retain such behavior inconsistently. A short verified trajectory therefore provides cleaner hindsight with less irrelevant context and lower teacher-side scoring cost.

Importantly, τ^{ref} is used only as privileged context for scoring the student's own on-policy actions, rather than as a demonstration target. The benefit of shortest-reference selection therefore comes from the quality of the hindsight signal, not from directly imitating a shorter solution. This selection rule can also be extended beyond binary success signals. In GRM-based settings, where different successful rollouts may receive different rewards, reference trajectories can be ranked by their rollout-level reward and filtered for both quality and efficiency. More generally, reference selection can incorporate any reliable trajectory-level signal that helps identify concise and high-quality solutions, while leaving the distillation target itself on-policy.

D Action Partition Validation

We manually inspect the action-level partition produced by PESD on 20 failed ALFWorld episodes sampled from the Qwen2.5-3B-Instruct policy at step 250. Each episode contains 50 replayable turns, yielding 1,000 annotated actions in total. Each action is labeled using the task, interaction history, current observation, admissible actions, and subsequent environment response, without access to the retrieved skill, reference trajectory, or PESD assignment.

Table 8: Manual annotations for 1,000 actions from 20 failed ALFWorld episodes.

Manual label	**Count**	**Fraction**
Correct	246	24.6%
Wrong	124	12.4%
Invalid	400	40.0%
Unclear	230	23.0%

An action is labeled *correct* only when it can be unambiguously identified as useful progress along a valid task-completion path. For compound or free-form outputs, we judge the first environment

command that is actually executed. An action is labeled *wrong* when it is executable but clearly makes no useful progress, targets the wrong object, skips a required transformation, or repeats an already completed operation. It is labeled *invalid* when no usable environment command is executed, for example when the environment returns `Nothing happens`. We use *unclear* when an action is locally plausible but its contribution to task completion cannot be determined reliably. In particular, once earlier mistakes have moved the episode away from a clear completion path, a locally reasonable action is labeled *unclear* unless its contribution to recovery is unambiguous. The *correct* label therefore denotes identifiable task progress rather than local syntactic validity. The final annotations contain 246 correct, 124 wrong, 400 invalid, and 230 unclear actions.

D.1 Default Partition

Table 9 compares the manual annotations with the PESD partition under the default consistency threshold $\rho = 0.6$. Most actions fall into $\mathcal{U}$ (665), while only 24 are assigned to the negative region $\mathcal{N}$. Among these 24 actions, 10 are wrong, 3 are invalid, 9 are unclear, and 2 are labeled correct. Thus, reverse supervision is applied to a small subset of failed-trajectory actions, with only 2 manually verified correct actions entering $\mathcal{N}$ in this sample.

Table 9: PESD partition versus manual annotations under the default $\rho = 0.6$.

	Correct	**Wrong**	**Invalid**	**Unclear**	**Total**
$\mathcal{P}$	56	11	209	35	311
$\mathcal{N}$	2	10	3	9	24
$\mathcal{U}$	188	103	188	186	665

The three regions define optimization roles rather than semantic action classes. $\mathcal{N}$ identifies actions eligible for conservative suppression, $\mathcal{U}$ leaves inconclusive behavior to RL, and $\mathcal{P}$ contains actions whose likelihood increases under the privileged context. The manual audit therefore evaluates the selectivity of the supervision rule, rather than treating $\mathcal{P}/\mathcal{N}/\mathcal{U}$ as correctness labels.

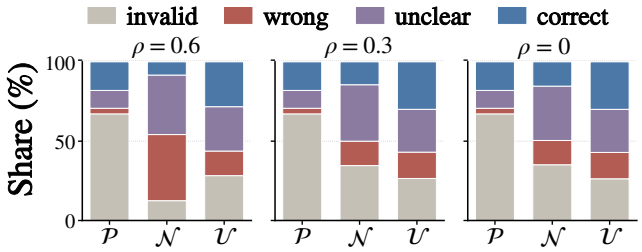


Figure 7: Manual annotations of actions assigned to $\mathcal{P}$, $\mathcal{N}$, and $\mathcal{U}$ under different consistency thresholds ρ.

D.2 Consistency Threshold

We next vary only the negative-consistency threshold ρ, keeping the action-level criterion $R_t < -\tau_-$ fixed. As shown in Figure 7 and table 10, lowering ρ from 0.6 to 0.3 expands $\mathcal{N}$ from 24 to 104 actions, while setting $\rho = 0$ increases it only slightly further to 111. At the same time, the number of manually labeled correct actions in $\mathcal{N}$ increases from 2 at $\rho = 0.6$ to 15 at $\rho = 0.3$, and to 17 when the consistency gate is removed.

The two correct actions selected at the default threshold are compound first-pass searches whose first executed command makes useful progress despite satisfying $c_t^- \geq 0.6$. The 13 additional correct actions introduced at $\rho = 0.3$ include useful navigation and manipulation steps, such as opening a relevant container, moving a required object to its destination, or navigating toward an appliance needed by the task. Lowering ρ further to 0 adds only seven more actions to $\mathcal{N}$, including two additional correct actions.

The default threshold yields the sparsest negative region and the lowest fraction of manually verified correct actions. Lowering ρ from 0.6 to 0.3 substantially expands $\mathcal{N}$ and increases the share of correct actions from 8.3% to 14.4%. Removing the consistency requirement adds only seven

Table 10: Effect of the consistency threshold on the negative region.

ρ	$\lvert\mathcal{N}\rvert$	**Wrong**	**Invalid**	**Unclear**	**Correct**
0.60	24	10/24 (41.7%)	3/24 (12.5%)	9/24 (37.5%)	**2/24 (8.3%)**
0.30	104	16/104 (15.4%)	36/104 (34.6%)	37/104 (35.6%)	15/104 (14.4%)
0.00	111	17/111 (15.3%)	39/111 (35.1%)	38/111 (34.2%)	17/111 (15.3%)

more actions, with a further small increase in correct assignments. We therefore use $\rho = 0.6$ as a conservative default and leave less consistent cases to $\mathcal{U}$ and the RL objective.

D.3 Positive Supervision on Failures

The manual audit indicates that $\mathcal{P}$ reflects support from the privileged context rather than semantic correctness. We therefore examine whether forward distillation on $\mathcal{P}$ contributes to downstream performance. We compare full PESD with two variants: *No Failure Distillation*, which removes all auxiliary supervision from failed trajectories, and *Reverse Only*, which retains unlikelihood updates on $\mathcal{N}$ but removes forward distillation on $\mathcal{P}$. The success branch and all other training settings are unchanged.

Table 11: Effect of failure-trajectory supervision on ALFWorld with Qwen2.5-3B-Instruct.

Failure-Trajectory Supervision	**Avg**
None	89.1
Reverse only	90.6
Forward + Reverse (PESD)	**92.2**

Removing all failure-side supervision yields an average of 89.1. Retaining only the reverse branch improves this to 90.6, while adding forward supervision on $\mathcal{P}$ further raises the average to 92.2. Thus, forward distillation on privileged-supported actions provides additional benefit beyond conservative suppression alone. This result also reinforces the intended interpretation of $\mathcal{P}/\mathcal{N}/\mathcal{U}$: the regions define how failed-trajectory actions are supervised, rather than serving as explicit labels of action quality.

E Datasets

ALFWorld. ALFWorld (Shridhar et al., 2020) is a text-based embodied benchmark in which an agent completes household tasks through multi-step interaction with the environment. We use the standard training split and evaluate six task families: pick-and-place (*Pick*), look-at-object-in-light (*Look*), clean-then-place (*Clean*), heat-then-place (*Heat*), cool-then-place (*Cool*), and pick-two-then-place (*Pick2*). These tasks require the agent to locate objects, manipulate them in the correct order, and complete long-horizon goals from textual observations. Following the SDAR evaluation protocol (Lu et al., 2026a), we report the main results on the 128-task subset of `valid_seen`. We additionally evaluate all 134 tasks from `valid_unseen` for unseen-environment generalization in Figure 2.

Search-QA. Search-QA (Jin et al., 2025) evaluates multi-turn information seeking with an external search tool. It combines seven question-answering datasets: Natural Question (NQ) (Kwiatkowski et al., 2019), TriviaQA (Joshi et al., 2017), PopQA (Mallen et al., 2023), HotpotQA (Yang et al., 2018), 2WikiMultiHopQA (Ho et al., 2020), MuSiQue (Trivedi et al., 2022), and Bamboogle (Press et al., 2023). The collection covers both direct retrieval and multi-hop questions that require evidence to be gathered across multiple search steps. At each turn, the agent may issue a search query, inspect the returned evidence, and continue searching before producing its final answer. We report the unweighted mean across the seven datasets. For periodic validation, we use up to 512 examples sampled uniformly across the seven sources. Alternative sample-level aggregation is discussed separately in Appendix A.

F EXPERIMENT DETAILS

We evaluate PESD on ALFWorld and Search-QA using Qwen2.5-3B-Instruct and Qwen3-1.7B. All reported runs use on-policy GRPO with rollout group size 8, learning rate 10^{-6}, maximum response length 512, validation before training, and experience-free evaluation. The PESD settings are fixed across benchmarks: success forward coefficient 0.01, failure forward coefficient 0.01, reverse coefficient 0.001, $(\tau_+, \tau_-, \rho) = (0.05, 0.20, 0.60)$, scheduled L_1–L_3 ladder with completed-episode context c^{traj}, revisit ratio 0.15, lexical applicability gate ($\gamma = 0.35$), and no environment fork. An LLM skill rewriter is not used. Failure distillation is disabled for the first 30 optimization steps and linearly warmed up over the following 20 steps. Its contribution is further modulated by the training-success EMA using thresholds of 0.03 and 0.10. For Qwen3, thinking mode is disabled so that generated responses follow the same environment-facing action interface.

F.1 ALFWORLD.

ALFWorld experiments use the TextWorld environment with a horizon of 50 interaction steps.

Table 12: ALFWorld training and evaluation settings.

Setting	Value
Environment	ALFWorld / TextWorld
Models	Qwen2.5-3B-Instruct, Qwen3-1.7B
Train batch / rollout group	16 / 8
Validation set	128 `valid_seen`; 134 `valid_unseen` tasks
Evaluation metric	pooled success on `valid_seen`
Max prompt / response length	2048 / 512
Environment horizon	50 steps
KL coefficient	0.01
Validation frequency	every 10 steps
Training budget	300 steps
SkillBank	ALFWorld SkillBank

Additional ALFWorld Results. We evaluate PESD over three independent runs on ALFWorld for both model scales. Table 13 reports the mean and standard deviation across runs.

Table 13: Additional ALFWorld results for PESD. Values are mean ± standard deviation over three independent runs.

Model	**Pick**	**Look**	**Clean**	**Heat**	**Cool**	**Pick2**	**Avg**
Qwen2.5-3B-Instruct	99.0±1.5	86.1±7.9	94.9±3.6	97.8±3.2	84.0±0.0	87.0±2.6	**92.2±1.3**
Qwen3-1.7B	79.2±8.9	41.7±0.0	66.7±1.8	71.1±3.1	73.3±3.8	44.5±12.0	**66.1±1.0**

The overall performance is stable across runs, with standard deviations of 1.3 and 1.0 for Qwen2.5-3B-Instruct and Qwen3-1.7B, respectively. Larger variation is concentrated in individual task families, while the aggregate performance remains consistent across training runs.

F.2 SEARCH-QA.

Search-QA uses a maximum of 5 search/action steps. Validation is reported as the unweighted mean over the seven data sources. The larger prompt budget accommodates retrieved passages and multi-hop evidence.

G ALGORITHM

Algorithm 1 summarizes the full PESD training procedure. At each update, the policy collects fresh on-policy trajectories together with a small fraction of revisited tasks. For each trajectory the teacher context $C_k = (c^{\text{traj}}, L_k)$ is rebuilt, and L_k is reduced when the same task–outcome pair is seen again. Successful trajectories use filtered forward distillation, whereas failed trajectories are partitioned into supported, rejected, and uncertain actions before applying signed distillation. RL is applied to all trajectories throughout training.

Table 14: Search-QA training and evaluation settings.

Setting	Value
Environment	Search-QA
Models	Qwen2.5-3B-Instruct, Qwen3-1.7B
Train batch / rollout group	128 / 8
Validation cap	512 examples
Full validation	51, 713 examples
Evaluation metric	seven-source mean success
Sources	NQ, TriviaQA, PopQA, HotpotQA, 2Wiki, MuSiQue, Bamboogle
Max prompt / response length	4096 / 512
Environment horizon	5 steps
KL coefficient	0.001
Validation frequency	every 20 steps
Training budget	300 steps
Search workers	64
SkillBank	Search-QA SkillBank

Algorithm 1 Progressive Experience Self-Distillation (PESD)

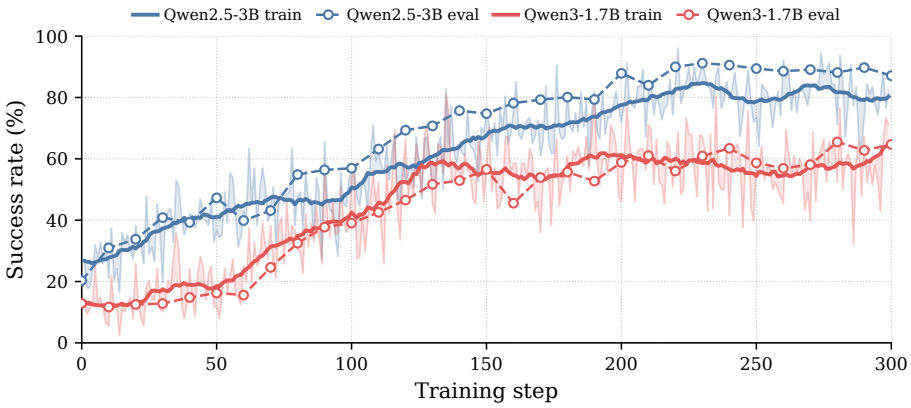
Require: Policy π_θ, task distribution $\mathcal{D}$, experience bank $\mathcal{B}$, reference cache $\mathcal{R}$, revisit buffer $\mathcal{M}$

for training step $u = 1, \ldots, U$ **do**
  Sample fresh and revisited tasks and collect on-policy trajectories
  Compute trajectory rewards and $\mathcal{L}_{\mathrm{GRPO}}$
  for each trajectory τ for task q **do**
    Retrieve $s^* \leftarrow \mathrm{Retrieve}(q, \tau, \mathcal{B})$ and form e^{skill} by the applicability gate
    if τ succeeds **then**
      Determine the active level k for $(q, \mathrm{success})$
      Build $C_k = (c^{\mathrm{traj}}, L_k)$
      Compute filtered forward distillation $\mathcal{L}_{\mathrm{succ}}$
    else
      Retrieve τ^{ref} from $\mathcal{R}$ if available
      Determine the active level k for $(q, \mathrm{failure})$
      Build $C_k = (c^{\mathrm{traj}}, L_k)$
      Score each action under privileged and original contexts
      Partition actions into $\mathcal{P}$, $\mathcal{N}$, and $\mathcal{U}$ using τ_+, τ_-, and ρ
      Compute forward loss on $\mathcal{P}$ and reverse loss on $\mathcal{N}$
    end if
    Add q to revisit buffer $\mathcal{M}$
  end for
  $\mathcal{L} \leftarrow \mathcal{L}_{\mathrm{GRPO}} + \lambda_{\mathrm{PESD}} \mathcal{L}_{\mathrm{PESD}}$
  Update θ using $\nabla_\theta \mathcal{L}$
end for
return π_θ

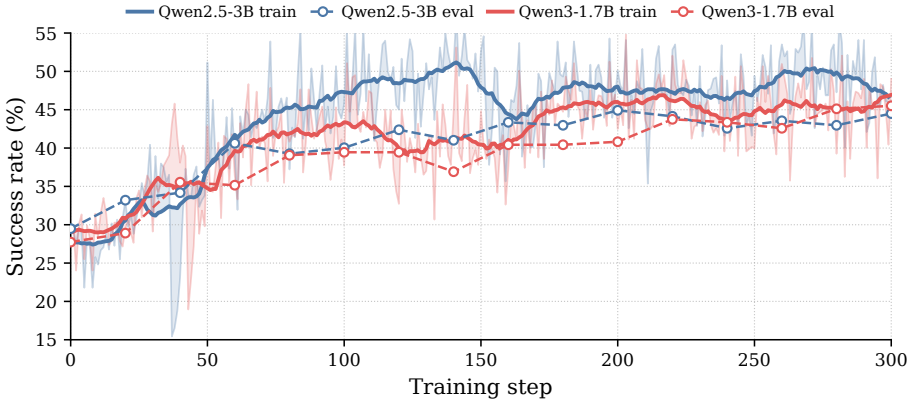
The privilege state is maintained independently for successful and failed outcomes of the same task. A newly observed failure starts from L_3 when a verified trajectory is available and otherwise from L_2; repeated visits reduce L_k toward L_1. Revisited tasks are rolled out again with the latest policy, so C_k is always built on current on-policy behavior. At evaluation time, the policy receives neither the completed episode as teacher context nor the scheduled ladder.

H Additional Training Dynamics

Figure 8 shows the training dynamics under a 300-step budget. On ALFWorld, both models improve over training, and Qwen2.5-3B-Instruct remains above Qwen3-1.7B after the first 100 steps. On Search-QA, most gains occur in the first half of training before performance levels off, with a smaller separation between backbones. In both settings, held-out performance remains stable toward the end of training.



(a) ALFWorld



(b) Search-QA

Figure 8: Training dynamics of PESD with Qwen2.5-3B-Instruct and Qwen3-1.7B. Solid curves show smoothed on-policy training success, with shaded regions indicating batch-level rates. Dashed curves with markers show held-out autonomous evaluation, measured every 10 steps on ALFWorld and every 20 steps on Search-QA. For Search-QA, each evaluation uses 512 queries sampled uniformly across the seven datasets.