COLLECT FOR COVERAGE, REPLAY WITH EVIDENCE: ADAPTIVE AND EFFICIENT ON-POLICY DISTILLATION FOR LONG-HORIZON AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Fully online on-policy distillation (OPD) is costly for long-horizon agents because each update depends on fresh environment rollouts. Prefix replay offers an efficient alternative by sampling current-student responses from stored histories, reducing idle time for long-tail rollouts and allowing each collected trajectory to support multiple updates. However, our analysis of the complete prefix replay pipeline identifies two limitations in long-horizon settings. In trajectory collection, teacher-only trajectories miss student-induced histories, while broader history coverage can reduce teacher reliability. In trajectory reuse, fixed position-based schedules cannot identify critical decisions that may occur at any trajectory depth. We introduce CARE-OPD (Collect for coverAge, Replay with Evidence) to address both limitations. Student-seeded teacher completions and response-level confidence gating expand history coverage while filtering unreliable updates. Hindsight replay allocation then uses downstream execution evidence to select consequential turns. Across long-horizon tasks Software Engineering and Deep Research, CARE-OPD outperforms online OPD and prior prefix replay baselines. On SWE-bench Verified, it raises task success to 59.4% while reducing end-to-end training time by 45% and environment actions by 44% relative to asynchronous online OPD.

1 INTRODUCTION

On-policy distillation (OPD) trains a student on its own responses while a stronger teacher supplies dense token-level targets (Gu et al., 2024; Agarwal et al., 2024). Fully online OPD for interactive agents executes the student in the environment (Yao et al., 2023), so training covers histories induced by the student's actions. This alignment becomes expensive over long horizons spanning tens or hundreds of agent turns. A rollout may search a repository, issue web queries, and wait for external tools, so trajectory durations can differ substantially. Synchronous training batches wait for their slowest trajectories, whereas asynchronous pipelines reduce synchronization idle time but still retain the full interaction cost and can introduce stale-policy data (Kang et al., 2026; Chen et al., 2026a). Long and heterogeneous rollouts therefore make training much slower than the model update alone.

Prefix replay offers an off-environment alternative. A separate collection stage records multi-turn trajectories from fixed policy snapshots. During training, the learner selects a stored prefix, generates a response of only one turn from the current student, and evaluates the teacher along that response. The generated response is not executed in the environment. Prefix replay therefore removes variable-length environment rollouts from the learner's critical path and allows one executed trajectory to support repeated current-student updates.

Prefix replay reduces training cost by constructing updates from stored histories rather than fresh environment rollouts. The quality of these updates depends on two design choices: **which histories to collect so that replay reflects situations the student may face while retaining reliable teacher supervision, and which turns to replay so that training focuses on decisions worth learning**. Concurrent work ReOPD collects teacher-generated histories and uses a monotone position-decay schedule to favor early prefixes (Liao et al., 2026). This design makes both parts of replay static: collection fixes the history source to the teacher, while allocation within each collected trajectory

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.

depends only on turn position. These choices are restrictive for long-horizon agents, where early actions can alter many later turns, and consequential decisions can occur at any depth. We address the resulting collection and reuse challenges in turn.

The first challenge is to pair student-relevant histories with confident teacher supervision. Teacher-only trajectories omit states reached after student errors, and this history mismatch compounds over long horizons (Ross et al., 2011; Li et al., 2026a). To cover situations that the student may face, a frozen student generates an initial segment before the teacher completes the trajectory. These student-induced histories make the replay contexts more relevant to the student, but they may fall outside the teacher's usual interaction distribution. We therefore pair student-seeded teacher completions with response-level confidence gating: collection broadens coverage toward student-induced histories, while the gate retains responses for which the teacher provides a confident target.

The second challenge is to identify which turns in a long trajectory are worth replaying. Long trajectories interleave routine interactions with consequential diagnoses, corrections, and verification decisions (Li et al., 2026c). Replaying every turn spends training computation on steps with limited learning value, while position decay can suppress valuable decisions that occur late in the trajectory (Liao et al., 2026). We instead allocate replay after observing the completed trajectory, when later execution feedback and the final outcome can support the value of earlier decisions. After each rollout, we append a selection query to the rollout teacher and reuse its trajectory KV cache to select these evidence-supported turns efficiently.

Together, these components form CARE-OPD (Collect for coverAge, Replay with Evidence). CARE-OPD addresses trajectory collection and reuse through reliability-controlled history coverage and hindsight replay allocation. Under matched hardware, CARE-OPD reduces end-to-end training time by 45% and executes 56% as many environment interactions as fully online OPD. On two long-horizon tasks SWE and Deep Research, it improves task success over all prefix replay and fully online OPD methods. We make three contributions:

(1) We analyze the complete prefix replay pipeline as two design problems: collecting student-relevant histories with reliable teacher supervision, and identifying which decisions in a long trajectory are worth replaying.

(2) We develop an adaptive replay method that addresses both problems. Student-seeded teacher completions cover histories induced by student behavior, while response-level confidence gating filters uncertain teacher supervision. Downstream execution evidence then directs replay toward consequential diagnoses, corrections, and verification decisions.

(3) We evaluate the complete training process on Software Engineering and Deep Research tasks. The method improves task performance over fully online OPD and prefix replay methods while reducing wall-clock time and environment interactions. Therefore, we establish prefix replay as a practical environment-decoupled training regime for long-horizon agents.

2 Related Work

2.1 Efficient Multi-Turn On-Policy Distillation

On-policy distillation (OPD) trains a student on its own responses using a teacher's next-token distribution (Gu et al., 2024; Agarwal et al., 2024). Recent extensions adapt OPD to multi-turn agents through teacher intervention, budget-aware rollout and loss allocation, hybrid objectives, and asynchronous rollout pipelines (Li et al., 2026b; Zhou et al., 2026c; Tan et al., 2026; Kang et al., 2026; Chen et al., 2026a). These methods retain live environment execution during student training.

Prefix replay collects histories independently and trains on current-student responses generated from stored prefixes, keeping environment execution off the training process. Concurrent work ReOPD reuses prefixes from teacher-generated trajectories and assigns each turn a sampling probability that decreases monotonically with its position in the trajectory, thereby favoring earlier turns (Liao et al., 2026). We find that this step-decay schedule cannot effectively transfer to long-horizon tasks spanning tens or hundreds of turns. We analyze the complete prefix replay pipeline for long-horizon tasks and identify the limitations in both stages, trajectory collection and reuse. We deal with both: student-seeded teacher completions expand history coverage under response-level confidence control, while downstream execution evidence allocates replay across the completed trajectory.

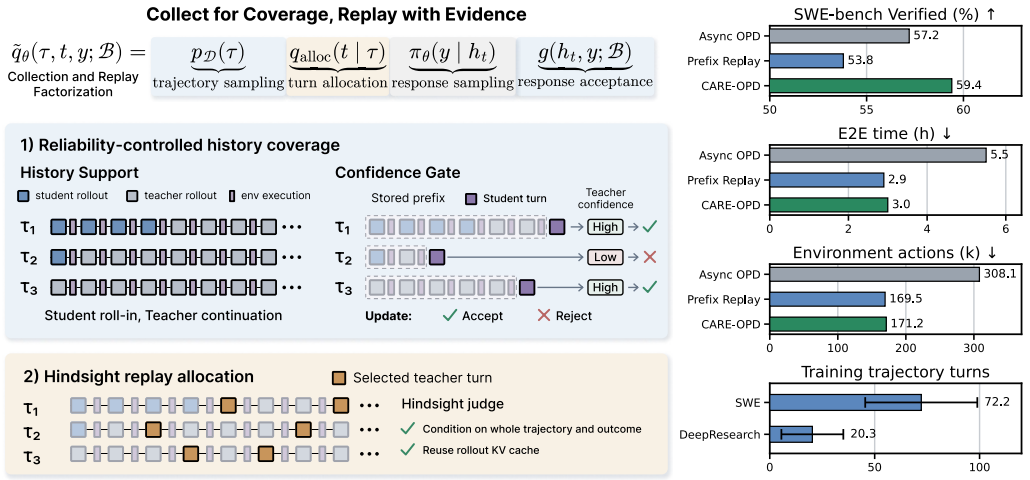

Figure 1: Overview of CARE-OPD. *Left:* Factorization of prefix replay in agent OPD (above). Method overview (below). *Right:* The plots compare SWE-bench Verified task success, end-to-end time, and environment actions on Qwen3.5 family. The bottom panel reports mean trajectory length with standard deviation during CARE-OPD training on SWE and Deep Research. Async OPD denotes online asynchronous one-turn distillation, and Prefix Replay denotes vanilla prefix replay.

2.2 Selecting Histories and Updates in OPD

OPD depends on both the teacher target and the histories where the teacher is queried. Fully online OPD uses student-induced histories, while Guided-OPD mixes teacher- and student-generated turns through an intervention curriculum (Li et al., 2026b). TrOPD restricts OPD to regions with reliable teacher supervision and uses teacher prefixes for off-policy guidance (Xing et al., 2026). These methods control the tension between student-relevant histories and teacher reliability through online intervention and trust regions. CARE-OPD instead constructs a replay pool with student-seeded teacher completions, then reassesses teacher confidence after the current student samples a response.

Several OPD methods also select or weight supervision within a trajectory. TurnOPD allocates rollout and loss budgets using turn-level statistics, SAGE-OPD selects teacher interventions from environment feedback and weights supervision by teacher confidence, and ATOD reweights turns using teacher–student disagreement and uncertainty (Zhou et al., 2026c;a; Tan et al., 2026). Other work masks or reweights OPD supervision using self-uncertainty, token teachability, or position-dependent teacher reliability (Ke et al., 2026; Wang et al., 2026; Liu et al., 2026). These rules derive their decisions from online trajectories, model distributions, or token position. Our replay setting adds an allocation decision over stored prefixes: evidence from the completed trajectory determines where the replay loss is evaluated, and teacher confidence on the newly sampled response determines whether that loss contributes an update.

3 Method

In this section, we first define agentic OPD and prefix replay, then analyze the whole pipeline and identify two limitations of static prefix policies. CARE-OPD addresses them through reliability-controlled history coverage and hindsight replay allocation.

3.1 Agentic OPD with Prefix Replay

OPD in agentic tasks. We consider tasks $x \sim p(x)$ in a long-horizon environment. At turn t, a policy π conditions on history $h_t = (x, o_1, a_1, \ldots, o_{t-1}, a_{t-1}, o_t)$ and samples a complete response $a_t \sim \pi(\cdot \mid h_t)$. The environment returns an observation o_{t+1}, which may be terminal. A rollout produces trajectory $\tau = (x, o_1, a_1, \ldots, o_T, a_T, o_{T+1})$.

Let N_τ be the number of student-generated tokens in τ, and let s_k denote the context before the k-th token. Let π_θ and π_T denote the current student and fixed teacher, respectively. Fully online agent OPD executes the current student to sample $\tau \sim p_\theta(\cdot \mid x)$, then minimizes the token-averaged reverse KL along the trajectory:

$$\mathcal{L}_{\mathrm{OPD}}(\theta) = \mathbb{E}_{\substack{x\sim p(x),\\ \tau\sim p_\theta(\cdot|x)}} \left[\frac{1}{N_\tau} \sum_{k=1}^{N_\tau} D_{\mathrm{KL}}(\pi_\theta(\cdot \mid s_k) \,\|\, \pi_T(\cdot \mid s_k)) \right]. \tag{1}$$

Each learner update therefore depends on a fresh rollout in the environment.

Prefix replay. Prefix replay removes this coupling between environment interaction and policy update. A separate collection stage generates trajectories from fixed policy snapshots and stores their interaction histories. During training, prefix replay truncates a stored trajectory at turn t to obtain h_t, then samples a response $y \sim \pi_\theta(\cdot \mid h_t)$. The teacher is evaluated along y, but the response is never executed. For a response of length L, the replay loss is

$$\mathcal{L}_{\mathrm{replay}}(\theta; h_t, y) = \frac{1}{L} \sum_{j=1}^{L} D_{\mathrm{KL}}(\pi_\theta(\cdot \mid h_t, y_{<j}) \,\|\, \pi_T(\cdot \mid h_t, y_{<j})). \tag{2}$$

One replay update generates a response instead of a trajectory, and one executed history can support repeated updates as the student changes. Prefix replay therefore removes environment latency from the learner's critical path. It still leaves open which histories are available and which candidate updates receive training computation.

3.2 Trajectory Collection and Reuse

We first analyze how trajectory collection and reuse determine which replay candidates contribute to student training in prefix replay. Let $\mathcal{D}$ denote the retained trajectory pool, h_t the prefix of τ before turn t, and $\mathcal{B}$ the current replay minibatch. The current student samples $y \sim \pi_\theta(\cdot \mid h_t)$ from this prefix. We decompose the unnormalized training mass of candidate (τ, t, y) as

$$\widetilde{q}_\theta(\tau, t, y; \mathcal{B}) = \underbrace{p_{\mathcal{D}}(\tau)}_{\text{trajectory sampling}} \underbrace{q_{\mathrm{alloc}}(t \mid \tau)}_{\text{turn allocation}} \underbrace{\pi_\theta(y \mid h_t)}_{\text{response sampling}} \underbrace{g(h_t, y; \mathcal{B})}_{\text{response acceptance}}. \tag{3}$$

This formulation exposes two limitations of static prefix policies in long-horizon tasks. First, teacher-only collection for $\mathcal{D}$ misses student-induced histories. Meanwhile, $p_{\mathcal{D}}$ determines the available histories but cannot ensure reliable teacher targets for the responses y, and broader history coverage can place the teacher in unfamiliar states. History coverage and response acceptance are therefore designed jointly. Second, q_{alloc} distributes replay across turns of different importance, where a fixed position rule cannot well identify critical turns to learn.

We address the first limitation by expanding teacher rollouts with student roll-ins, and gate the responses with teacher confidence. We address the second with an allocation rule conditioned on the completed trajectory. CARE-OPD uses the whole trajectory and execution evidence to decide where replay occurs. We detail the two components below.

3.3 Reliability-Controlled History Coverage

Prefix replay is on-policy for the sampled response but off-policy for its stored history. A teacher-only pool omits states caused by earlier student actions, and one early deviation can change many later observations. We extend the candidate history distribution with student-seeded teacher completions. A frozen student snapshot generates an initial trajectory segment, after which the teacher continues from the resulting history.

For each trajectory i, we draw $z_i \sim \mathrm{Bernoulli}(\alpha)$, where α is the student-seeded trajectory fraction. If $z_i = 0$, the teacher generates the full trajectory. If $z_i = 1$, we sample the student roll-in length uniformly as $K_i \sim \mathrm{Uniform}\{1, \ldots, K_{\max}\}$. The student acts through turn K_i, and the teacher completes the trajectory from turn $K_i + 1$. Let π_{θ_0} denote the frozen student used for collection. The behavior policy is

$$\pi_i^{\mathrm{col}}(\cdot \mid h_t) = \begin{cases} \pi_{\theta_0}(\cdot \mid h_t), & z_i = 1 \text{ and } t \le K_i, \\ \pi_T(\cdot \mid h_t), & \text{otherwise.} \end{cases} \tag{4}$$

The roll-in rate changes the source of collected histories. Student turns remain as context and never become replay targets, while only turns from the teacher continuation are eligible for allocation. If the student completes the task during the roll-in, we discard the trajectory because it contains no teacher continuation for replay.

Expanded history coverage does not guarantee a reliable update. We observe more low-confidence evaluations on the student responses near the switch points of student-seeded trajectories, as analyzed in Section 4.4. We therefore decide acceptance after sampling the response. For a candidate prefix h and response $y \sim \pi_\theta(\cdot \mid h)$, let $p_j^T = \pi_T(\cdot \mid h, y_{<j})$ at token position j. Inspired by the turn-level confidence weighting in SAGE-OPD (Zhou et al., 2026a), we summarize teacher certainty using the mean top-token probability:

$$P_T(h, y) = \frac{1}{L} \sum_{j=1}^{L} \max_{v \in \mathcal{V}} p_j^T(v), \tag{5}$$

where L is the response length and $\mathcal{V}$ is the token vocabulary. For rejection fraction $\gamma \in [0, 1]$, let $\tau_\gamma(\mathcal{B})$ be the γ-quantile of confidence scores in minibatch $\mathcal{B}$. We accept the update when

$$g_\gamma(h, y; \mathcal{B}) = \mathbb{I}[P_T(h, y) \geq \tau_\gamma(\mathcal{B})]\,. \tag{6}$$

The gate rejects the lowest-confidence fraction γ and keeps the effective batch size predictable. It treats confidence as a proxy for teacher certainty rather than response correctness. A rejected response contributes no gradient, but its prefix remains available for later replay. Because OPD already computes the teacher logits in Equation 5, the gate adds little inference cost.

3.4 Hindsight Replay Allocation

Reliability-controlled coverage determines where replay can occur, but long trajectories still contain many candidate turns with limited training value. We allocate replay after each trajectory terminates, when later observations and the final outcome are available. A model-based judge receives the task, complete trajectory, execution feedback, outcome, and a fixed selection budget. It selects teacher-generated turns whose diagnosis, correction, or verification is supported by later evidence.

For trajectory i, let $\mathcal{E}_i$ denote the eligible teacher-generated turn indices. Given selection fraction r, the judge returns a subset $\mathcal{S}_i \subseteq \mathcal{E}_i$ of size $m_i = \lceil r|\mathcal{E}_i| \rceil$. This defines

$$q_{\text{alloc}}(t \mid \tau_i) = \frac{\mathbb{I}[t \in \mathcal{S}_i]}{m_i}. \tag{7}$$

The fixed within-trajectory fraction avoids score calibration across tasks. During training, we first sample a trajectory and then one of its selected turns, preventing long trajectories from receiving extra weight solely because they contain more turns.

We assign each turn a temporary identifier and append one selection query after the rollout terminates. The query reuses the rollout teacher's trajectory KV cache. We remove the temporary identifiers and selector output before replay, so the student receives the original prefix without future evidence.

3.5 Training Procedure and Cost

Algorithm 1 summarizes the complete CARE-OPD procedure. Environment interactions occur only during pool construction, and hindsight replay allocation runs once per collected trajectory. The selected prefixes can support repeated updates as the student changes. Training therefore retains response-level on-policy generation without placing long environment rollouts on the learner's critical path.

Algorithm 1 CARE-OPD: collection, hindsight allocation, and OPD training.

Require: Collection task set $\mathcal{X}_{\text{col}}$; π_T, π_{θ_0}, α, $K_{\max}$, r, γ; optimization steps U, batch size B, learning rate η
Ensure: Trained student π_θ

Stage I: Collection and hindsight allocation

$\mathcal{D} \leftarrow \varnothing$
for $x \in \mathcal{X}_{\text{col}}$ **do**
  Sample $z \sim \text{Bernoulli}(\alpha)$
  If $z = 1$, sample $K \sim \text{Uniform}\{1, \ldots, K_{\max}\}$ ▷ student roll-in length
  Execute π^{col} from Equation 4 on x to obtain τ; let $\mathcal{E}$ be its teacher-generated turns
  Discard τ and **continue** if verification fails or $\mathcal{E} = \varnothing$ ▷ retain verified completions
  $\mathcal{S} \leftarrow \textsc{HindsightSelect}(\pi_T, x, \tau, \mathcal{E}, \lceil r|\mathcal{E}| \rceil)$ ▷ allocate turns using downstream evidence
  $\mathcal{D} \leftarrow \mathcal{D} \cup \{(\tau, \mathcal{S})\}$
end for

Stage II: Replay training

$\theta \leftarrow \theta_0$
for $u = 1, \ldots, U$ **do**
  Sample minibatch $\{(\tau_b, \mathcal{S}_b)\}_{b=1}^{B}$ uniformly from $\mathcal{D}$ ▷ $\mathcal{S}_b$: hindsight-selected turns of τ_b
  Sample $t_b \sim \text{Uniform}(\mathcal{S}_b)$ and extract h_b at t_b for all b ▷ sample one prefix per trajectory
  Sample $y_b \sim \pi_\theta(\cdot \mid h_b)$ for all b
  Set $\ell_b \leftarrow \mathcal{L}_{\text{replay}}(\theta; h_b, y_b)$ and $c_b \leftarrow P_T(h_b, y_b)$ for all b ▷ calculate loss and teacher confidence
  $q_\gamma \leftarrow \textsc{Quantile}(\{c_b\}_{b=1}^{B}, \gamma)$; $\mathcal{A} \leftarrow \{b : c_b \geq q_\gamma\}$ ▷ reject bottom-γ confidence
  $\theta \leftarrow \theta - \eta \nabla_\theta \dfrac{1}{|\mathcal{A}|} \displaystyle\sum_{b \in \mathcal{A}} \ell_b$
end for
return π_θ

4 Experiments

4.1 Experimental Setup

Data and benchmarks. We study two long-horizon agentic tasks: Software Engineering (SWE) and Deep Research. For **Software Engineering**, we partition SWE-Gym (Pan et al., 2025) into a training set and a validation set of 100 tasks. We collect all training trajectories from the training set and evaluate task success on SWE-bench Verified (Jimenez et al., 2024). For **Deep Research**, as high-quality data is important, we use a dataset curated from public sources with a quality-control pipeline, detailed in Appendix A. We evaluate answer accuracy on BrowseComp-Plus (Chen et al., 2026b), using Qwen3-Embedding-8B as the retriever and gpt-oss-20b as the judge.

Agent scaffolds. For SWE, we use OpenHands as the agent scaffold and retain its default system prompt and tools (Wang et al., 2025). For Deep Research, we customize the Uni-Agent scaffold (Ding et al., 2026) and provide three tools: `search`, `open`, and `find`. SFT and OPD training both use a 128K context limit and a 150-turn limit.

Implementation details. For OPD training, we use 4 GPU nodes, where 1 node is for the teacher, 2 nodes are for student rollouts, and 1 node is for student updates. We optimize for 60 steps. Appendix B lists the complete training configuration.

Models. We conduct experiments using two model families, Qwen3.5 and Qwen3. For **Qwen3.5**, Qwen3.5-9B is the student and Qwen3.5-27B is the teacher, with thinking enabled. For **Qwen3**, Qwen3-8B is the student and Qwen3-Coder-30B-A3B-Instruct is the teacher. Qwen3-8B and Qwen3-Coder-30B-A3B-Instruct use different native tool-call formats, making direct OPD ineffective, so we first cold-start Qwen3-8B as detailed in Appendix C. We report the cold-start model as the Qwen3 initial student and use it to initialize every Qwen3 training method. For Qwen3 on BrowseComp-Plus, the teacher has a small performance gap relative to the initial student and Appendix D reports the scores, so we omit experiments for this setting.

Baselines. All methods within each model family start from the same initial checkpoint. The **SFT** baseline behavior-clones successful teacher trajectories retained by rejection sampling. For Deep Research, we generate candidate SFT trajectories from a fixed sample of 3,840 tasks, matching the task budget of each OPD run. **Fully online OPD** executes the current student in the environment and applies the OPD loss. **ReOPD** uses teacher-generated histories and position-decayed replay

with the best-performing value in our sweep, $\kappa = 0.9$, and Appendix E reports the full sweep. **Vanilla prefix replay** uses teacher-generated histories and samples eligible turns uniformly, without hindsight replay allocation or confidence gating.

4.2 Main Results

Table 1: Task success (%) on SWE-bench Verified and answer accuracy (%) on BrowseComp-Plus. All training methods start from the same initial student. We omit Qwen3 Deep Research experiments because its candidate teacher exceeds the initial student by only 4.6 points on BrowseComp-Plus, leaving limited room for distillation; Appendix D reports this assessment.

	Qwen3.5 family		*Qwen3 family*
Method	SWE-bench Verified↑	BrowseComp-Plus↑	SWE-bench Verified↑
Reference models			
Teacher	67.2	57.8	43.0
Initial student	53.2	42.5	10.8
Offline teacher imitation			
SFT	54.8	46.1	17.8
Online OPD			
Full trajectory (all turns)	56.4	<u>47.2</u>	<u>22.0</u>
One random turn	<u>57.6</u>	47.0	20.4
Prefix replay			
Vanilla prefix replay	53.8	45.8	17.2
ReOPD (position decay)	54.0	45.4	18.6
CARE-OPD (ours)	**59.4**	**49.5**	**22.8**

CARE-OPD outperforms both online OPD and existing prefix replay methods. This result holds across all three settings, showing that adaptive history construction and replay allocation allow environment-decoupled replay to surpass training on fresh online trajectories. The following paragraphs analyze these comparisons in detail.

One-turn online OPD is comparable to full-trajectory OPD, but fresh student histories matter. The two online variants differ by only 1.2, 0.2, and 1.6 points across the three settings, with no consistent winner. On Qwen3.5 SWE-bench Verified, trajectories average 84.9 turns, yet random-turn OPD scores 1.2 points higher. Less reliable targets after accumulated student errors in long trajectories may limit the value of full-trajectory distillation. However, vanilla prefix replay trails one-turn online OPD by 3.8, 1.2, and 3.2 points. This contrast points to fixed teacher-generated prefixes and absent environment execution, rather than one-turn supervision itself: replay cannot reach later histories induced by the current student's decisions. Section 4.3 below shows the trade-off: vanilla prefix replay is substantially cheaper but less effective.

Position-based replay allocation yields limited gains in long trajectories. Relative to vanilla prefix replay, ReOPD changes performance by $+0.2$, -0.4, and $+1.4$ points. Its monotone schedule assumes that useful supervision concentrates near the trajectory start. Long-horizon agents also make consequential diagnoses, corrections, and verification decisions at later turns, so fixed decay can suppress useful examples. Section 4.5 examines this limitation directly. Together, the online and replay comparisons show that more supervision is not consistently better: full-trajectory OPD does not reliably beat one-turn OPD, and position decay does not reliably beat uniform replay. CARE-OPD instead adapts history coverage and selects turns from trajectory evidence, consistent with its gains over both replay variants.

4.3 Does Prefix Replay Improve End-to-End Training Efficiency?

We compare synchronous online OPD, asynchronous online OPD, vanilla prefix replay, and CARE-OPD under identical GPU, environment worker, and batch size configurations. Appendix B reports the detailed hyperparameters. End-to-end time includes every stage needed to obtain the trained student. For CARE-OPD, this consists of trajectory collection, hindsight replay allocation, and replay

training. Environment actions count all attempted trajectories, including unsuccessful collection attempts.

Table 2: SWE end-to-end training efficiency and task success. Task success is reported in percent. Parentheses give ratios relative to synchronous online OPD with one distilled turn.

Method	E2E time↓	Env. actions↓	Task success↑
Sync OPD, one turn	7.5 h (1.00×)	309.9k (1.00×)	<u>57.6</u>
Async OPD, one turn	5.5 h (0.73×)	308.1k (0.99×)	57.2
Async OPD, all turns	5.8 h (0.77×)	314.8k (1.02×)	56.4
Vanilla prefix replay	**2.9** h (**0.39**×)	**169.5**k (**0.55**×)	53.8
CARE-OPD	<u>3.0</u> h (0.40×)	<u>171.2</u>k (0.55×)	**59.4**

Table 2 shows that asynchronous one-turn OPD reduces end-to-end time by 27% relative to synchronous one-turn OPD while achieving comparable task success (57.2% vs. 57.6%). However, it executes nearly the same number of environment actions (308.1k vs. 309.9k) and retains the latency tail of full environment rollouts. By decoupling environment interaction from student training, vanilla prefix replay and CARE-OPD reduce end-to-end time by 61% and 60% relative to synchronous one-turn OPD, and by 47% and 45% relative to asynchronous one-turn OPD. Both replay methods also execute approximately 45% fewer environment actions than either online baseline because each collected trajectory supports repeated prefix updates. CARE-OPD takes 3.0 hours and executes 171.2k environment actions, only 0.1 hours and 1.7k actions more than vanilla prefix replay, while improving task success by 5.6 points. Its evidence-adaptive components therefore add little cost to vanilla prefix replay.

4.4 How Do History Coverage and Response Acceptance Interact?

Student-seeded collection broadens history coverage, but sampled responses on the resulting prefixes can expose the teacher to states outside its usual interaction distribution. We first measure teacher uncertainty around the roll-in switch, then compare training outcomes across history pools and gate choices.

Measure teacher uncertainty. We examine teacher uncertainty immediately after the student-to-teacher switch on collected trajectories with student roll-in. For each response, we report the mean top-token probability P_T from Equation 5 and the token-averaged teacher entropy $H_T = -\frac{1}{L}\sum_{j=1}^{L}\sum_{v\in\mathcal{V}} p_j^T(v)\log p_j^T(v)$.

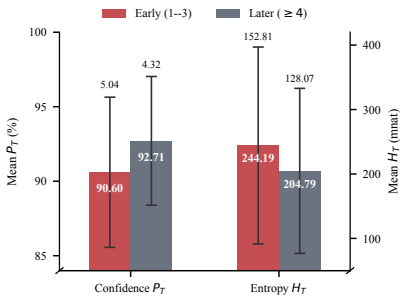

Figure 2: Teacher uncertainty after student-to-teacher switch points.

Figure 2 summarizes the shift in the first three teacher turns after the switch. Relative to later turns, the first three teacher turns has a lower mean P_T by 2.11 points and raise mean H_T by 39.40 mnat. Their standard deviations also exceed the later values. The early low-confidence ($< 90\%$) and high-entropy (> 400 mnat) rates are 35.13% and 16.50%, compared with 26.24% and 9.32% over all post-switch turns. This localized shift motivates response-level reliability control.

Confidence gating complements expanded history coverage. We next measure how history coverage and response acceptance affect downstream training. Table 3 forms a 2×2 comparison between teacher-only and student-seeded trajectory pools, with or without confidence gating. The student-seeded pool uses our default fraction $\alpha = 0.1$. Student roll-in with gating reaches 57.0 points, a 3.2-point gain over the baseline. This gain exceeds the 2.0-point sum of the two standalone gains by 1.2 points, indicating a positive interaction. Confidence gating is therefore more useful on the expanded history support, consistent with the elevated teacher uncertainty after student roll-in.

Table 3: Interaction between history coverage and response acceptance on SWE-bench Verified. We report task success (%).

History pool	No confidence gate	Confidence gate
Teacher-only	53.8	54.6
Student-seeded	55.0	**57.0**

4.5 HINDSIGHT REPLAY ALLOCATION IMPROVES PREFIX REPLAY

We vary replay allocation strategies. Table 4 compares five allocation rules. Uniform replay samples all eligible turns, while random subset selection retains 40% of the turns in each trajectory without using their content. Best position decay uses the best-performing ReOPD setting in our sweep ($\kappa = 0.9$) in Appendix E. The separate critic uses a stronger critic model Qwen3.5-122B-A10B for hindsight replay allocation, whereas our judge appends one selection query to the rollout teacher and reuses its trajectory KV cache. Both model-based selectors use the same 40% per-trajectory budget.

Table 4: Controlled comparison of replay-allocation rules on SWE-bench Verified.

Allocation rule	Retained turns	Task success (%)
Uniform replay	100%	53.8
Random subset	40%	53.6
Best position decay (ReOPD)	weighted	54.0
Separate critic	40%	**57.6**
Teacher hindsight	40%	57.2

Trajectory evidence drives the allocation gain. The hindsight allocation in CARE-OPD improves task success by 3.4 points over uniform replay and by 3.6 points over the matched random subset. Random subset selection is comparable with uniform replay, so retaining fewer turns alone does not improve performance. Best position decay improves over uniform replay by 0.2 points but remains 3.2 points below our allocation. These results indicate that selecting turns using trajectory evidence accounts for the gain, while a monotone preference for early turns yields only a marginal improvement on these long trajectories.

Hindsight allocation preserves useful late turns. Figure 3 visualizes the normalized replay mass. The step-deday rule assigns most replay mass to the first few turns, whereas hindsight replay allocation spreads replay across the first several dozen turns and retains a longer tail. These later turns include corrections, recovery actions, and verification steps supported by subsequent execution evidence. Their value depends on what happened in the trajectory rather than on a fixed depth schedule.

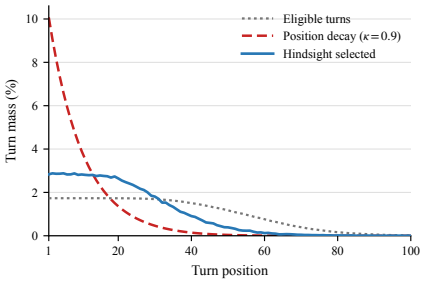

Figure 3: Replay mass by turn position over the first 100 turns.

The rollout teacher nearly matches a separate critic. The separate critic reaches 57.6%, only 0.4 points above our 57.2%. Reusing the rollout teacher therefore nearly matches the stronger independent critic while avoiding an additional full-trajectory prefill. Appendices F and E report the hyperparameter and position-decay sweeps, respectively.

5 LIMITATIONS AND CONCLUSION

Limitations. Although our student-seeded collection broadens history coverage, it uses a frozen initial student and bounded roll-ins, while the student changes during training. The replay pool may therefore become less representative over time. Periodically refreshing the pool could reduce this mismatch at additional environment cost. Replayed responses are not executed, leaving their validity and environmental consequences unobserved. Teacher confidence measures distributional certainty rather than response correctness. Future work could selectively execute uncertain responses or develop execution-aware correctness estimators.

Conclusion. On-policy distillation is expensive for long-horizon agents because each update typically requires a fresh environment rollout. We study prefix replay as an efficient alternative, and our analysis identifies two central design problems: trajectory collection and trajectory reuse. We introduce CARE-OPD, which addresses trajectory collection with reliability-controlled history coverage and trajectory reuse with evidence-based allocation. CARE-OPD outperforms online OPD and prior prefix replay methods across long-horizon tasks SWE and Deep Research. On SWE-bench Verified, it achieves 59.4% task success with 45% less end-to-end training time and 44% fewer environment actions than asynchronous online OPD. Further analyses show that adaptive collection and reuse make prefix replay an effective and efficient training approach for long-horizon agents.

AI USE DISCLOSURE

We used generative AI for both synthetic data and manuscript preparation. As detailed in Appendix A, DeepSeek-V4-Flash audited Deep Research query–answer pairs for answerability and support from currently accessible evidence, and gpt-oss-20b evaluated answers on BrowseComp-Plus. We also used generative AI tools to edit parts of the manuscript for clarity. The authors reviewed and revised all AI-assisted text. We take full responsibility for the content of this work.

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*, 2024. URL https://arxiv.org/abs/2306.13649.

Xianwei Chen, Shimin Zhang, and Jibin Wu. f-OPD: Stabilizing long-horizon on-policy distillation with freshness-aware control. *arXiv preprint arXiv:2605.17862*, 2026a. URL https://arxiv.org/abs/2605.17862.

Zijian Chen, Xueguang Ma, Shengyao Zhuang, Ping Nie, Kai Zou, Sahel Sharifymoghaddam, Andrew Liu, Joshua Green, Kshama Patel, Ruoxi Meng, Mingyi Su, Yanxi Li, Haoran Hong, Xinyu Shi, Xuye Liu, Hosna Oyarhoseini, Nandan Thakur, Crystina Zhang, Luyu Gao, Wenhu Chen, and Jimmy Lin. BrowseComp-Plus: A fair and disentangled evaluation benchmark for deep search agents. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics*, pp. 22349–22370, 2026b. doi: 10.18653/v1/2026.acl-long.1023. URL https://aclanthology.org/2026.acl-long.1023/.

Zheng Chu, Xiao Wang, Jack Hong, Huiming Fan, Yuqi Huang, Yue Yang, Guohai Xu, Chenxiao Zhao, Cheng Xiang, Shengchao Hu, Dongdong Kuang, Ming Liu, Bing Qin, and Xing Yu. REDSearcher: A scalable and cost-efficient framework for long-horizon search agents. *arXiv preprint arXiv:2602.14234*, 2026. URL https://arxiv.org/abs/2602.14234.

DeepSeek-AI. DeepSeek V4 Technical Documentation. Technical report, 2026. URL https://fe-static.deepseek.com/chat/transparency/deepseek-V4-model-card-EN.pdf.

Yuyang Ding, Bo Wen, Xubo Cao, Zhiqiang Zhai, Guangming Sheng, Xibin Wu, Juntao Li, Min Zhang, and Uni-Agent Contributors. Uni-Agent: Build, run, and train agents at scale. GitHub repository, 2026. URL https://github.com/verl-project/uni-agent. Supervisors: Xibin Wu and Juntao Li.

Yuwen Du, Rui Ye, Shuo Tang, Xinyu Zhu, Yijun Lu, Yuzhu Cai, and Siheng Chen. OpenSeeker: Democratizing frontier search agents by fully open-sourcing training data. *arXiv preprint arXiv:2603.15594*, 2026. URL https://arxiv.org/abs/2603.15594.

Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. MiniLLM: Knowledge distillation of large language models. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/8ac015d409635f196f9e3e9dcfb9a94e-Abstract-Conference.html.

Naman Jain, Jaskirat Singh, Manish Shetty, Liang Zheng, Koushik Sen, and Ion Stoica. R2E-Gym: Procedural environments and hybrid verifiers for scaling open-weights SWE agents. *arXiv preprint arXiv:2504.07164*, 2025. URL https://arxiv.org/abs/2504.07164.

Carlos E. Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. SWE-bench: Can language models resolve real-world GitHub issues? In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/edac78c3e300629acfe6cbe9ca88fb84-Abstract-Conference.html.

Wonjun Kang, Kevin Galim, Seunghyuk Oh, Minjun Kang, Sanghyun Park, Donghoon Kim, Minjae Lee, Minseo Kim, Rishabh Tiwari, Yuchen Zeng, Hyung Il Koo, and Kangwook Lee. AsyncOPD: How stale can on-policy distillation be? *arXiv preprint arXiv:2606.24143*, 2026. URL https://arxiv.org/abs/2606.24143.

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. URL https://arxiv.org/abs/2605.13255.

Changhao Li, Rushi Qiang, Jiawei Huang, Chenxiao Gao, Chao Zhang, Niao He, and Bo Dai. Revisiting DAgger in the era of LLM-agents. *arXiv preprint arXiv:2605.12913*, 2026a. URL https://arxiv.org/abs/2605.12913.

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*, 2026b. URL https://arxiv.org/abs/2606.15912.

Mukai Li, Qingcheng Zeng, Tianqing Fang, Zhenwen Liang, Linfeng Song, Qi Liu, Haitao Mi, and Dong Yu. Verified critical step optimization for LLM agents. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 39627–39639. Association for Computational Linguistics, 2026c. doi: 10.18653/v1/2026.findings-acl.1974. URL https://aclanthology.org/2026.findings-acl.1974/.

Zhuofeng Li, Dongfu Jiang, Xueguang Ma, Haoxiang Zhang, Ping Nie, Yuyu Zhang, Kai Zou, Jianwen Xie, Yu Zhang, and Wenhu Chen. OpenResearcher: A fully open pipeline for long-horizon deep research trajectory synthesis. *arXiv preprint arXiv:2603.20278*, 2026d. URL https://arxiv.org/abs/2603.20278.

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. URL https://arxiv.org/abs/2607.04763.

Xiaogeng Liu, Xinyan Wang, Yingzi Ma, Yechao Zhang, and Chaowei Xiao. When are teacher tokens reliable? position-weighted on-policy self-distillation for reasoning. *arXiv preprint arXiv:2605.21606*, 2026. URL https://arxiv.org/abs/2605.21606.

Kyle Montgomery. Public deep research SFT dataset releases. Hugging Face datasets, 2026. URL https://huggingface.co/kylemontgomery.

Jiayi Pan, Xingyao Wang, Graham Neubig, Navdeep Jaitly, Heng Ji, Alane Suhr, and Yizhe Zhang. Training software engineering agents and verifiers with SWE-Gym. In *Proceedings of the 42nd International Conference on Machine Learning*, 2025. URL https://arxiv.org/abs/2412.21139.

Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pp. 627–635. JMLR Workshop and Conference Proceedings, 2011.

Qitai Tan, Zefang Zong, Mo Li, Yipeng Shi, Yang Li, and Peng Chen. ATOD: Annealed turn-aware on-policy distillation for multi-turn agentic tasks. *arXiv preprint arXiv:2606.27814*, 2026. URL https://arxiv.org/abs/2606.27814.

Xingyao Wang, Boxuan Li, Yufan Song, Frank F. Xu, Xiangru Tang, Mingchen Zhuge, Jiayi Pan, Yueqi Song, Bowen Li, Jaskirat Singh, Hoang H. Tran, Fuqiang Li, Ren Ma, Mingzhang Zheng, Bill Qian, Yanjun Shao, Niklas Muennighoff, Yizhe Zhang, Binyuan Hui, Junyang Lin, Robert Brennan, Hao Peng, Heng Ji, and Graham Neubig. OpenHands: An open platform for AI software developers as generalist agents. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=OJd3ayDDoF.

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. URL https://arxiv.org/abs/2605.26844.

Xingrun Xing, Haoqing Wang, Boyan Gao, Ziheng Li, and Yehui Tang. Trust region on-policy distillation. *arXiv preprint arXiv:2606.01249*, 2026. URL https://arxiv.org/abs/2606.01249.

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

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. URL https://arxiv.org/abs/2606.19659.

Yuhang Zhou, Kai Zheng, Qiguang Chen, Mengkang Hu, Qingfeng Sun, Can Xu, and Jingjing Chen. OffSeeker: Online reinforcement learning is not all you need for deep research agents. *arXiv preprint arXiv:2601.18467*, 2026b. URL https://arxiv.org/abs/2601.18467.

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*, 2026c. URL https://arxiv.org/abs/2607.05804.

APPENDIX

A DEEP RESEARCH DATASET CONSTRUCTION

We construct the Deep Research training-task pool from five public sources. We combine OpenResearcher (Li et al., 2026d), DeepForge from OffSeeker (Zhou et al., 2026b), OpenSeeker (Du et al., 2026), REDSearcher (Chu et al., 2026) and public Deep Research SFT collections released by Kyle Montgomery (Montgomery, 2026). Source releases may contain complete agent trajectories, but we retain only the query and its reference answer from each example. We discard every source trajectory and never use it as a training target.

We normalize each query with Unicode NFKC normalization, case folding, and whitespace collapsing. We then merge the sources and deduplicate queries across all sources. This process yields 75,372 queries.

Because the source corpora were collected with different pipelines and at different times, their query–answer pairs vary in quality and current verifiability. We use DeepSeek-V4-Flash (DeepSeek-AI, 2026) as a task-level answerability probe in a common live-web environment. The probe can call `search`, `open`, and `find`, while web search and reading use Serper.[1]. We retain a candidate when the probe's final answer is consistent with the source reference and supported by evidence available in the current web environment. This pass screens for ambiguous queries, stale reference answers, and questions whose supporting evidence is no longer accessible. It also reduces variation caused by source-specific retrieval environments. After filtering, the remaining 16,855 tasks form the curated training dataset for Deep Research.

B TRAINING HYPERPARAMETERS

All OPD-based methods use four GPU nodes, each equipped with eight NVIDIA H200 GPUs. We assign one node to teacher inference, two nodes to student rollout generation, and one node to student updates. Across these methods, we use a learning rate of 10^{-6}, a batch size of 64, and 60 optimization steps.

For asynchronous OPD, we set the maximum policy staleness to 4 learner updates. We observe that staleness values at most 4 have little effect on final task performance, whereas larger values reduced performance. For SFT, we use a learning rate of 10^{-5}, a batch size of 64, and train for three epochs. Table 5 summarizes these settings.

Table 5: Training hardware and optimization hyperparameters.

Configuration	OPD	SFT
Learning rate	10^{-6}	10^{-5}
Batch size	64	64
Training duration	60 steps	3 epochs
Maximum staleness	4 (asynchronous OPD only)	–

C QWEN3 COLD START

Qwen3-8B and Qwen3-Coder-30B-A3B-Instruct use different native tool-call formats. The raw student therefore produces invalid calls under the Coder format, making direct OPD ineffective. We randomly sample 500 tasks from R2E-Gym-Subset (Jain et al., 2025), whose repositories do not overlap with SWE-bench, and use Qwen3-Coder-30B-A3B-Instruct to generate candidate trajectories. We retain successful trajectories through rejection sampling and perform SFT on Qwen3-8B using the retained data. The raw Qwen3-8B model scores 8.6% on SWE-bench Verified, while the cold-start model scores 10.8%. We report the cold-start model as the Qwen3 initial student and use it to initialize every Qwen3 training method.

[1] `https://serper.dev/`

D QWEN3 DEEP-RESEARCH ASSESSMENT

We evaluated the Qwen3 teacher and the same cold-start initial student on BrowseComp-Plus. Table 6 shows that Qwen3-Coder-30B-A3B-Instruct reaches 17.3% accuracy, while Qwen3-8B reaches 12.7%. The teacher is only 4.6 percentage points higher than the student. Thus, we exclude this setting from our experiments.

Table 6: Answer accuracy on BrowseComp-Plus for the candidate Qwen3 teacher and initial student.

Model	Role	Accuracy (%)
Qwen3-Coder-30B-A3B-Instruct	Candidate teacher	17.3
Qwen3-8B	Initial student	12.7

E SENSITIVITY TO THE REOPD POSITION-DECAY PARAMETER

We sweep the position-decay parameter κ on SWE-bench Verified using the same collection and attempted-update budgets across settings. For the eligible turn set $\mathcal{E}_\tau$ of trajectory τ, we apply the ReOPD schedule to the absolute turn index and sample turns according to

$$q_{\text{alloc}}(t \mid \tau) = \frac{\kappa^t}{\sum_{j \in \mathcal{E}_\tau} \kappa^j}, \qquad t \in \mathcal{E}_\tau. \tag{8}$$

At $\kappa = 1.0$, this schedule assigns equal probability to all eligible turns (vanilla prefix replay).

Table 7: Sensitivity to the position-decay parameter κ on SWE-bench Verified. The setting $\kappa = 1$ is uniform replay.

κ	0.5	0.7	0.9	1.0
Task success (%)	53.0	53.8	**54.0**	53.8

Table 7 shows that $\kappa \in \{0.7, 0.9, 1.0\}$ yields similar task success, within a 0.2-point range. Strong decay degrades performance: reducing κ to 0.5 lowers task success to 53.0%, 1.0 point below the best position-decay setting. At $\kappa = 0.5$, the replay weight halves at every turn, so a prefix ten turns later receives only $0.5^{10} \approx 0.1\%$ of the weight assigned to the earlier prefix. This rapid decay concentrates replay on the start of a long trajectory and leaves little training mass for later turns, which likely contributes to the observed decline.

F SENSITIVITY TO KEY HYPERPARAMETERS

We evaluate three hyperparameters on SWE-bench Verified: the student-seeded trajectory fraction α, rejection fraction γ, and hindsight selection fraction r. Each sweep varies one hyperparameter, and holds the remaining settings fixed. The selected settings are $\alpha = 0.1$, $\gamma = 0.1$, and $r = 0.4$. We do not tune the maximum student roll-in length $K_{\max}$. Instead, we set it to the mean trajectory length observed during collection so that the roll-in range scales with the task horizon. For Qwen3.5, this rule gives $K_{\max} = 72$ on SWE and $K_{\max} = 20$ on Deep Research.

Performance is stable across the tested student-seeding fractions. Task success is 59.4%, 59.2%, and 58.6% at $\alpha = 0.1$, 0.3, and 0.5, respectively. This 0.8-point range suggests that CARE-OPD is relatively insensitive to α over the tested interval. We use $\alpha = 0.1$, which gives the highest task success while requiring student roll-ins for only 10% of the collected trajectories.

Rejecting a small low-confidence fraction is sufficient. Task success decreases markedly as γ increases. Because the gate removes the lowest-confidence fraction of each minibatch, increasing γ progressively discards more candidate updates. This decline suggests that most of the benefit comes from filtering a small set of potentially unreliable updates, while stronger rejection removes useful supervision.

Table 8: Sensitivity of CARE-OPD on SWE-bench Verified. Each block reports the sweep settings and corresponding task success for one hyperparameter. Bold marks the settings used in the final method.

Hyperparameter	Sweep setting	Task success (%)
Student-seeded fraction α	**0.1**	**59.4**
	0.3	59.2
	0.5	58.6
Rejection fraction γ	**0.1**	**59.4**
	0.3	58.8
	0.5	57.8
Hindsight fraction r	0.2	57.6
	0.4	**59.4**
	0.6	59.0
	0.8	58.4

Hindsight allocation benefits from a moderate selection budget. Selecting 20% of the eligible turns yields 57.6% task success, while $r = 0.4$ and $r = 0.6$ reach 59.4% and 59.0%, respectively. Performance decreases slightly to 58.4% at $r = 0.8$. With $r = 0.2$, the judge selects few turns and therefore covers a narrower set of trajectory states. We also observe that small selection fractions concentrate selected turns near trajectory starts, which can omit later corrections, recovery actions, and verification steps.