BRANCH-GRAFTED ON-POLICY DISTILLATION FOR AGENTS SOLVING CAPTURE-THE-FLAG CHALLENGES

Anonymous authors
Paper under double-blind review

ABSTRACT

On-policy distillation (OPD) trains a student policy using teacher supervision on trajectories generated by the student, thereby reducing the distribution mismatch between offline supervision and the states encountered during deployment. For multi-turn agents, we study how to make fuller use of teacher supervision at interaction histories already visited by the student. Standard OPD follows the response sampled by the student, while alternative teacher responses from the same history are not directly used for distillation. We introduce BRANCH-GRAFTED ON-POLICY DISTILLATION (BG-OPD), which augments student trajectories with single-turn teacher-generated response branches at selected interaction histories. BG-OPD uses teacher–student residual disagreement to allocate branch supervision, and distills the resulting teacher responses without executing their proposed actions in the environment. We further show that the branch objective can be interpreted as a turn-level approximation to residual-weighted teacher supervision and characterize the corresponding approximation error. We evaluate BG-OPD on interactive cybersecurity tasks, where long-horizon hypothesis testing and tool use make alternative response paths particularly important. To support on-policy training at scale, we construct 1,370 containerized CTF environments whose solvability is verified through independently executed solutions. Across Cybench, NYU CTF Bench, and XBOW, BG-OPD consistently outperforms offline SFT and existing OPD baselines for both 4B and 9B students, improving mean task success from 13.20% to 21.01% over the strongest baseline for the 9B model and from 7.70% to 14.76% for the 4B model. Ablations further attribute the improvement to teacher-generated branches and disagreement-based branch selection.

1 INTRODUCTION

On-policy distillation (OPD) transfers capabilities from stronger language models to smaller models by applying teacher supervision to trajectories generated by the student policy (Agarwal et al., 2024). This is particularly useful for multi-turn agents, where model actions determine subsequent environment states and observations, causing the distribution of interaction histories to evolve with the student policy. Recent work has therefore adapted OPD to multi-turn settings through rollout curricula, adaptive horizon control, and selective supervision (Wang et al., 2026; Zhou et al., 2026b;a; Chen et al., 2026). Complementary to these advances, we study how to make fuller use of teacher supervision at interaction histories already visited by the student. In standard OPD, supervision follows the response sampled by the student. At a visited history h_t, the teacher may generate a different response, but that response is not directly used for distillation. This motivates extending supervision at student-visited histories beyond the sampled response without requiring additional environment trajectories.

This limitation is especially relevant to capture-the-flag (CTF) agents, which repeatedly formulate hypotheses, invoke tools, and revise their behavior from environment feedback (Shao et al., 2024; Zhang et al., 2025; Ji et al., 2025). An unsuccessful interaction may provide only partial evidence against the current hypothesis, allowing an unproductive solution direction to persist across multiple turns. Standard OPD continues to supervise the sampled response prefixes in such trajectories, while alternative hypotheses or actions available to the teacher remain outside the supervised prefixes. Explicitly executing these alternatives would broaden coverage but require additional environment rollouts, increasing the cost of online training.

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.

We introduce **Branch-Grafted On-Policy Distillation (BG-OPD)**, which augments student trajectories with teacher-generated responses at selected student-visited histories. BG-OPD scores each turn using teacher-positive residual probability mass and samples branch locations in proportion to these scores. At a selected history, the teacher generates a single alternative response that is used for distillation but whose proposed action is not executed. Training combines reverse-KL supervision on the original student trajectory with forward-KL supervision on the grafted teacher branches, extending supervision beyond student-sampled prefixes without additional environment interaction. We further interpret the branch objective as a turn-level approximation to residual-weighted teacher supervision and characterize the approximation error induced by replacing response-specific residual weights with turn-level scores.

Large-scale on-policy training additionally requires executable environments with verifiable solutions. We therefore develop an evidence-guided reconstruction pipeline that builds or repairs target environments and solver scripts from public CTF artifacts, verifies each candidate through isolated execution, and iteratively repairs failed reconstructions. Using this pipeline, we construct **1,370 containerized CTF environments**, each retained only after an independently executed solver recovers a valid flag and after screening for solution leakage and evaluation overlap.

We evaluate BG-OPD with Qwen3.5-4B and Qwen3.5-9B students distilled from a fixed Qwen3.6-27B teacher on Cybench (Zhang et al., 2025), NYU CTF Bench (Shao et al., 2024), and XBOW (XBOW Engineering, 2024). BG-OPD consistently outperforms Offline SFT and existing OPD baselines across both model scales and all three benchmarks. For the 9B student, mean task success improves from 13.20% for the strongest OPD baseline to 21.01%; for the 4B student, it improves from 7.70% to 14.76%. Controlled experiments further show that teacher-generated branches outperform applying the same additional forward-KL supervision to original or resampled student responses, while our weighted sampling outperforms random and deterministic turn selection.

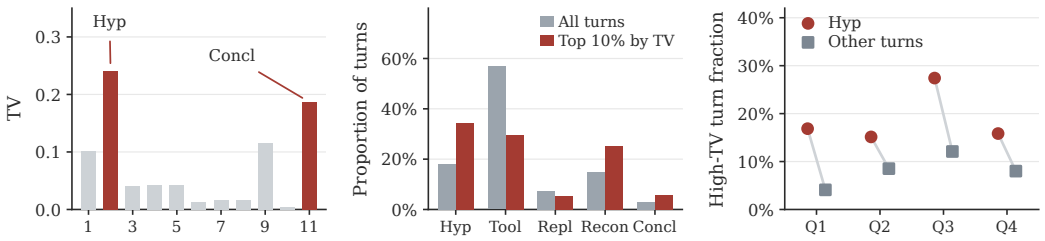
Our main contributions are:

- We propose **BG-OPD**, which grafts single-turn teacher responses onto student trajectories without executing branch actions, with a turn-level residual-weighting analysis.
- We construct 1,370 verifiably solvable, containerized CTF environments through evidence-guided reconstruction and in-loop verification.
- Across three CTF benchmarks and two model scales, BG-OPD consistently outperforms Offline SFT and all evaluated OPD baselines.

2 Problem Setup and Analysis

2.1 Preliminaries

Capture-the-flag (CTF) tasks. CTF challenges are self-contained cybersecurity tasks in which a solver recovers a hidden string, called a *flag*, as evidence of completing the challenge (Zhang et al., 2025). They span categories such as web security, binary exploitation, reverse engineering, cryptography, and forensics. A task provides a description, optional attachments (e.g., source code, binaries, or encrypted data), and, when required, an address for a target service. An agent uses its local workspace to inspect files, execute scripts, and interact with the target, using command outputs and service responses to guide its solution (Zhuo et al., 2026). A separate verifier checks the submitted flag against the expected value or validation rule for the current environment instance; Figure 8 in Appendix C illustrates this interaction and verification process. Formally, for a task x drawn from the task distribution $\mathcal{D}$, the agent proceeds through a sequence of tool-mediated interactions. Before generating its response at turn t, the agent observes the interaction history $h_t = (o_0, y_0, o_1, \ldots, y_{t-1}, o_t)$, where o_0 contains the initial task information and the remaining elements record the agent's previous responses and the resulting environment observations. Conditioned on h_t, the agent generates a response $y_t = (z_t, a_t)$, consisting of reasoning z_t followed by a single action a_t (Yao et al., 2022). The environment executes a_t and returns an observation o_{t+1}, yielding the updated history $h_{t+1} = (h_t, y_t, o_{t+1})$. The process ends when the agent submits a flag or exhausts the turn budget H, yielding a trajectory $\tau = (o_0, y_0, \ldots, y_{T-1}, o_T)$ of $T \leq H$ turns.



(a) Turn-level disagreement (b) Behavioral composition (c) Controlling for response length

Figure 1: **Turn-level teacher–student disagreement in CTF trajectories.** (a) Disagreement varies across an trajectory, with peaks at hypothesis formation and conclusion. (b) Hypothesis-forming turns are overrepresented among the global top 10% by TV. (c) This enrichment persists within response-length quartiles.

On-policy distillation (OPD). OPD is increasingly used to transfer capabilities from large models to smaller students and to consolidate capabilities across models(Yang et al., 2025; Team et al., 2026; Li et al., 2026b; Yoon et al., 2026). Given a teacher policy π_ϕ with fixed parameters ϕ and a student policy π_θ with trainable parameters θ, both sharing a vocabulary $\mathcal{V}$, OPD trains the student to match the teacher's token distributions along student-generated trajectories. The reverse-KL objective is

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

where $\pi_\theta(\cdot \mid h_t)$ and $\pi_\phi(\cdot \mid h_t)$ are distributions over the response y_t. By the chain rule, Eq. (1) decomposes exactly into per-token KL terms. Let N_t count the supervised response tokens in turn t. Each token k is conditioned on the interaction history h_t and the preceding response tokens $y_{t,<k}$. In practice we use *teacher-supported top-K reverse KL*:

$$\mathcal{L}_{\mathrm{OPD}}(\theta) = \mathbb{E}_{\tau\sim\pi_\theta}\left[\sum_{t=0}^{T-1}\sum_{k=1}^{N_t}\sum_{v\in\mathcal{S}_K(h_t,y_{t,<k})} \widetilde{\pi}_\theta(v \mid h_t, y_{t,<k}) \log \frac{\widetilde{\pi}_\theta(v \mid h_t, y_{t,<k})}{\widetilde{\pi}_\phi(v \mid h_t, y_{t,<k})}\right], \tag{2}$$

where $\mathcal{S}_K(h_t, y_{t,<k}) \subseteq \mathcal{V}$ contains the K highest-probability teacher tokens given h_t and $y_{t,<k}$. The distributions $\widetilde{\pi}_\theta$ and $\widetilde{\pi}_\phi$ are obtained by renormalizing the respective policies on this support. Restricting the loss to the teacher's top-K tokens stabilizes the training signal while avoiding the computational overhead of full-vocabulary supervision.

2.2 Motivation

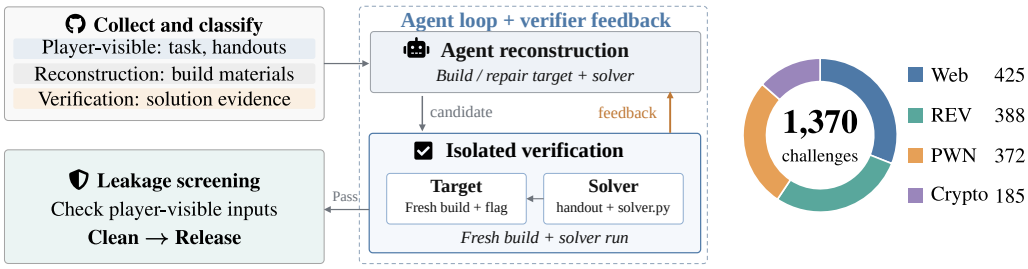
Vanilla OPD provides teacher supervision only along student-sampled trajectories, limiting opportunities to incorporate teacher guidance beyond the student's current rollout distribution. To characterize how teacher–student disagreement varies across turns in these trajectories, we define a TV score for each turn t in a student-generated trajectory by averaging, across its N_t response-token positions, the positive teacher–student probability residual summed over $\mathcal{S}_K$:

$$\widehat{\mathrm{TV}}_t = \frac{1}{N_t}\sum_{k=1}^{N_t}\sum_{v\in\mathcal{S}_K(h_t,y_{t,<k})} \big[\pi_\phi(v \mid h_t, y_{t,<k}) - \pi_\theta(v \mid h_t, y_{t,<k})\big]_+, \tag{3}$$

where $[u]_+ = \max(u, 0)$.

Figure 1(a) shows an illustrative vanilla OPD trajectory with TV peaks at turns labeled as hypothesis formation and conclusion.

For the aggregate analysis, we collect 28,815 interaction turns from vanilla OPD trajectories. We use DeepSeek-V4-Flash (DeepSeek-AI, 2026) to classify these turns into five categories: hypothesis formation (Hyp), tool execution (Tool), replanning (Repl), reconnaissance (Recon), and conclusion (Concl). *Reconnaissance* gathers information about the target, including its files, services, and environment. *Hypothesis formation* proposes a concrete vulnerability or solution path and identifies evidence that could test it. *Tool execution* carries out or tests an existing path through commands, scripts, or service interactions. *Replanning* revises or abandons the current path in response to new evidence. *Conclusion* reports the final flag or task outcome and summarizes the evidence supporting



(a) Construction and verification **(b) Environment categories**

Figure 2: CTF environment reconstruction pipeline and category distribution.

that result. Across these labeled turns, hypothesis formation accounts for 17.9% of all turns but 34.2% of the global top 10% by TV; the corresponding shares for tool execution are 56.8% and 29.6% (Figure 1(b)). Reconnaissance and conclusion turns are also enriched. To test whether the enrichment of hypothesis-forming turns is explained by response length, we repeat the comparison within each response-length quartile. Hypothesis-forming turns remain more likely than other turns to fall in the global top 10% by TV (Figure 1(c)).

High-TV turns are more likely to involve hypothesis formation or conclusion than turns overall. This pattern is consistent with the expectation that the teacher and student differ more when proposing a solution path or forming a final answer, supporting TV as a reasonable disagreement measure.

3 Method

We first present our CTF environment construction and verification pipeline (Section 3.1), and then introduce BG-OPD (Section 3.2), which grafts teacher-generated response branches onto student trajectories to extend teacher supervision beyond student-sampled response prefixes.

3.1 CTF Environment Construction and Verification

Public CTF materials often lack complete deployments or executable solutions. This limits opportunities for agents to obtain learning feedback through exploration and interaction with running environments. We therefore use the three-stage reconstruction and validation pipeline shown in Figure 2(a). **(1) Collection and classification.** We collect and classify publicly available CTF artifacts hosted on GitHub. These artifacts include challenge descriptions and handouts as well as source code and deployment files. We also collect available writeups and exploits. **(2) Reconstruction and verification.** An agent first uses these materials to build or repair a Docker target and its solver. A separate validator then builds and deploys the target from a clean state before running the solver in isolation. If the target fails to build or the solver encounters a runtime error or fails to recover a valid flag, the validator returns the failure to the agent for repair. This build–verify–repair cycle continues until validation succeeds. **(3) Screening and release.** We screen candidates that recover a valid flag for player-side leakage and benchmark overlap. Only candidates that pass both checks are retained. Implementation details are provided in Appendix A.2. Table 1 summarizes the scale and verification requirements of our environment collection alongside CTF-Dojo (Zhuo et al., 2025). Our pipeline feeds validation failures back to the construction agent for repair and requires an independently executed solver to recover a valid flag for every retained environment. Compared with validation workflows that rely on manual testing, this automated loop provides uniform per-task coverage and supports scalable verification.

Table 1: **Comparison with CTF-Dojo.**

	CTF-Dojo (Zhuo et al., 2025)	**Ours**
Environments	658	**1,370**
Verification	Post-hoc	In-loop
Per-task solvability	Not reported	Required

3.2 Branch-Grafted On-Policy Distillation

At each visited history, a student rollout provides only one response, leaving alternative response prefixes outside its distillation supervision. Motivated by the diagnostic analysis in Section 2.2, we

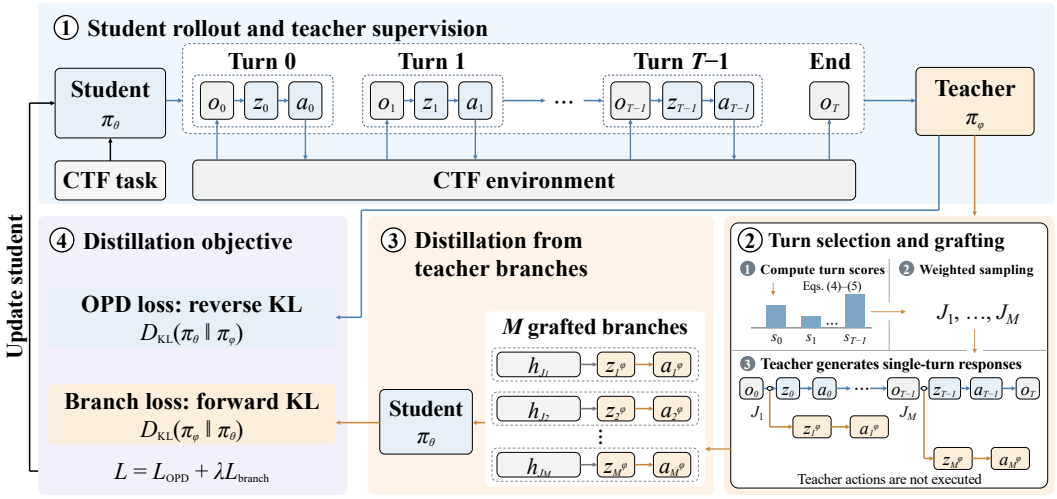


Figure 3: **Overview of BG-OPD.** (1) The student generates a complete trajectory. Token-level reverse KL against the teacher is then computed using Eq. (2). (2) M turns are sampled with probabilities proportional to teacher–student disagreement scores using Eqs. (4)–(6). The teacher generates single-turn responses from the history preceding each selected turn to form grafted branches, without executing teacher actions. (3) The student is supervised on these grafted branches using Eqs. (7)–(8). (4) The two supervision signals are combined using Eq. (9) to update the student.

propose grafting teacher continuations onto student trajectories at selected turns. Each branch starts from the history preceding its selected turn and provides additional supervision while preserving the original student trajectory (Figure 3).

Turn selection for grafting. Let $\pi_{\theta_{\mathrm{old}}}$ denote the student policy used to collect trajectories. For token k of the student response y_t, we compute the *teacher-positive residual mass* d_{tk}:

$$d_{tk} = \sum_{v \in \mathcal{S}_K(h_t, y_{t,<k})} \left[\pi_\phi(v \mid h_t, y_{t,<k}) - \pi_{\theta_{\mathrm{old}}}(v \mid h_t, y_{t,<k})\right]_+. \tag{4}$$

Scoring reuses the raw teacher probabilities computed for Equation (2), requiring no additional teacher forward passes. For each turn with $N_t > 0$, we obtain its score s_t by averaging these residuals over its response tokens:

$$s_t = \frac{1}{N_t} \sum_{k=1}^{N_t} d_{tk}. \tag{5}$$

This is the diagnostic statistic $\widehat{\mathrm{TV}}_t$ in Equation (3), evaluated under the rollout policy $\pi_{\theta_{\mathrm{old}}}$. For one trajectory, let $W = \sum_t s_t$ be the total turn score and $M \geq 1$ the number of teacher branches sampled when $W > 0$. The turn-sampling rule is

$$P(t) = \frac{s_t}{W} \quad (W > 0), \qquad J_m \sim P, \quad m = 1, \ldots, M. \tag{6}$$

Here $P(t)$ is the probability of selecting turn t, and J_m is the turn index selected for branch m. When $W = 0$, we retain the original trajectory without grafting. Otherwise, we draw these indices independently with replacement and independently sample a single-turn teacher response $y_m^\phi \sim \pi_\phi(\cdot \mid h_{J_m})$ from each selected turn's preceding history. Each response includes reasoning and an action, forming a branch alongside the original student response. The action proposed by the teacher is not executed in the environment. This response is incorporated into the student's branch distillation objective as additional supervision.

Branch distillation objective. For a teacher response $y^\phi \sim \pi_\phi(\cdot \mid h_t)$ at a selected turn t, let $N > 0$ be the number of supervised tokens in y^ϕ. We match the teacher's token distributions using teacher-supported top-K forward KL, averaged over these positions and conditioned on h_t and the

teacher response prefix $y^{\phi}_{<k}$:

$$\ell(\theta; h_t, y^{\phi}) = \frac{1}{N}\sum_{k=1}^{N} D_{\mathrm{KL}}\big(\widetilde{\pi}_{\phi}(\cdot \mid h_t, y^{\phi}_{<k}) \,\|\, \widetilde{\pi}_{\theta}(\cdot \mid h_t, y^{\phi}_{<k})\big). \tag{7}$$

For a trajectory with T valid turns, we average its M branch losses and scale them by W/T:

$$\mathcal{L}_{\mathrm{branch}}(\theta) = \mathbb{E}\left[\frac{W}{T}\frac{1}{M}\sum_{m=1}^{M} \ell(\theta; h_{J_m}, y^{\phi}_{m})\right]. \tag{8}$$

The expectation covers rollouts $\tau \sim \pi_{\theta_{\mathrm{old}}}$, turn selection, and teacher response generation. The contribution is defined as zero when $W = 0$. The factor W/T retains the residual-mass scale of the additional supervision, which vanishes as the full student and teacher distributions approach agreement. The overall training objective combines the two loss terms:

$$\mathcal{L}(\theta) = \mathcal{L}_{\mathrm{OPD}}(\theta) + \lambda \mathcal{L}_{\mathrm{branch}}(\theta). \tag{9}$$

Here $\lambda \geq 0$ controls the strength of branch supervision. The OPD term provides supervision along the student's own trajectories, while the branch term extends teacher guidance beyond student rollouts.

Understanding how teacher branches complement OPD. An ideal distillation signal should cover both the student's own trajectory and teacher-preferred branches that the student undersamples. At history $h = h_t$, the teacher-positive residual defines the ideal additional supervision weight:

$$[\pi_{\phi}(y \mid h) - \pi_{\theta_{\mathrm{old}}}(y \mid h)]_{+} = \pi_{\phi}(y \mid h) w(y), \tag{10}$$

where $w(y) = [1 - \pi_{\theta_{\mathrm{old}}}(y \mid h)/\pi_{\phi}(y \mid h)]_{+}$, set to zero when $\pi_{\phi}(y \mid h) = 0$. This ideal supplement can be realized by sampling teacher responses and weighting each by $w(y)$. Our method retains teacher sampling but replaces $w(y)$ with the turn score $s = s_t$, yielding a turn-level approximation. With $\ell(y) = \ell(\theta; h, y)$, the ideal loss $\mathbb{E}_{\pi_{\phi}}[w\ell]$ becomes the proxy $s\mathbb{E}_{\pi_{\phi}}[\ell]$. Expectations and covariance are under $y \sim \pi_{\phi}(\cdot \mid h)$. The sampling and scaling in Equation (8) give an unbiased estimate of this proxy aggregated across turns.

Proposition 1 (Residual-weighting error decomposition)**.** *Let* $\Delta = D_{\mathrm{TV}}(\pi_{\theta_{\mathrm{old}}}(\cdot \mid h), \pi_{\phi}(\cdot \mid h))$. *For fixed* θ *and finite loss expectations, the ideal residual loss and the turn-level proxy satisfy*

$$\mathbb{E}_{\pi_{\phi}}[w\ell] - s\mathbb{E}_{\pi_{\phi}}[\ell] = \mathrm{Cov}_{\pi_{\phi}}(w, \ell) + (\Delta - s)\mathbb{E}_{\pi_{\phi}}[\ell]. \tag{11}$$

The two terms capture response-specific weighting effects and the mismatch between turn scores and total response residual mass. Our branch objective thus serves as a tractable turn-level surrogate for residual-weighted teacher supervision, allocating additional supervision to turns with larger residual scores. The proofs and the connection to exact residual branching are provided in Appendix B.

4 Experiments

4.1 Experimental Setup

Benchmarks. We evaluate on Cybench (Zhang et al., 2025), NYU CTF Bench (Shao et al., 2024), and XBOW (XBOW Engineering, 2024). Cybench contains professional-level CTF challenges, NYU CTF Bench collects tasks from CSAW competitions, and XBOW focuses on web vulnerabilities. A task is successful if the agent recovers a valid flag within the interaction budget. We report task success rates for each benchmark.

Models and baselines. We use Qwen3.5-4B and Qwen3.5-9B (Qwen Team, 2026a) as students and Qwen3.6-27B (Qwen Team, 2026b) as the fixed teacher. Students learn through online interaction with our reconstructed CTF environments. We compare BG-OPD with vanilla OPD, TCOD-F2B (Wang et al., 2026), and TurnOPD (Zhou et al., 2026b).

Table 2: **Performance of BG-OPD and baselines on three CTF benchmarks.**

Method	Task Success Rate (%) ↑			
	Cybench	NYU CTF	XBOW	Avg
Qwen3.6-27B (Teacher)	42.50	37.69	47.12	42.43
Qwen3.5-9B (Student)				
Base student	7.50	2.51	9.62	6.54
Offline SFT	10.00	5.53	12.50	9.34
Vanilla OPD	15.00	6.53	16.35	12.63
TCOD-F2B	10.00	1.51	5.77	5.76
TurnOPD	17.50	10.55	11.54	13.20
Branch-Grafted OPD	**22.50**	**14.57**	**25.96**	**21.01**
Qwen3.5-4B (Student)				
Base student	2.50	0.50	0.96	1.32
Offline SFT	5.00	3.52	4.81	4.44
Vanilla OPD	10.00	2.51	10.58	7.70
TCOD-F2B	12.50	1.51	0.96	4.99
TurnOPD	7.50	6.03	8.65	7.39
Branch-Grafted OPD	**15.00**	**10.05**	**19.23**	**14.76**

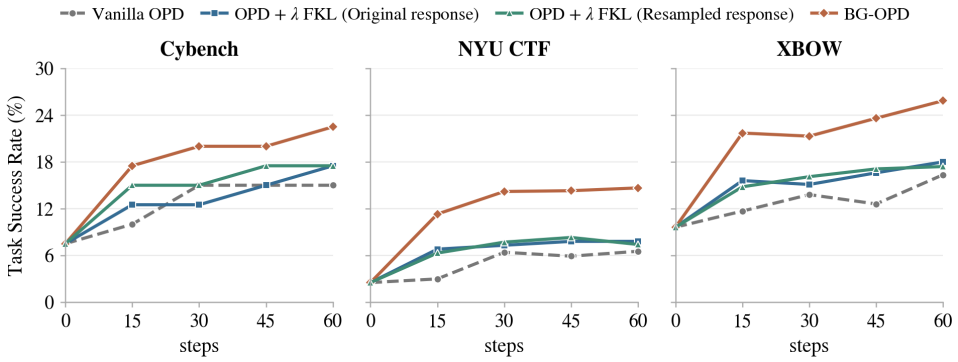


Figure 4: **Effect of grafting teacher-generated branches.** *Original response* and *Resampled response* apply the same forward-KL supervision to existing and newly sampled student responses, respectively, at the selected turns. Both controls improve over vanilla OPD, while grafting teacher-generated responses yields larger gains.

4.2 Main Results

BG-OPD outperforms both Offline SFT and all OPD baselines. Across model scales, it improves the average task success rate over Offline SFT by approximately 10%–12% and over the strongest OPD baselines by approximately 7%–8%.

Comparison with Offline SFT. Offline SFT trains each base student on successful teacher-generated trajectories retained by rejection sampling. It improves both base students but trails vanilla OPD on average at both scales. This suggests that students acquire teacher capabilities more effectively through environment interaction and exploration with environmental feedback and dense teacher supervision than through training on fixed offline data.

Comparison with OPD Baselines. BG-OPD consistently outperforms vanilla OPD, TCOD-F2B, and TurnOPD across all three benchmarks and both model scales. TCOD-F2B progressively extends the student rollout horizon, yet underperforms vanilla OPD on average at both scales, suggesting that deeper rollouts alone do not consistently improve performance. TurnOPD adapts rollout depth and rebalances KL supervision across turns, improving NYU CTF performance at both scales but yielding mixed gains overall. BG-OPD uses teacher–student disagreement to select turns for grafting teacher-generated responses at histories encountered by the student. These branches expose alternative hypotheses and actions while preserving supervision on the original trajectories, broadening the response prefixes available for distillation.

4.3 Analysis of BG-OPD

Effect of grafting teacher-generated branches. We compare BG-OPD with vanilla OPD and two FKL controls. The three branch-based runs use the same turn-selection strategy, $M = 8$ branches per trajectory, and a branch loss weight of $\lambda = 0.2$ (Figure 4). At each selected turn, *Original response* reuses the student's existing response, *Resampled response* draws a new single-turn response from the student, and BG-OPD grafts a single-turn response generated by the teacher. The three FKL variants use the teacher's token distributions as targets without executing branch actions. Both student-side controls improve over vanilla OPD by the end of training, suggesting that further alignment with the teacher through FKL is beneficial. The gains remain modest when supervision follows student-sampled response prefixes, and resampling offers no consistent improvement over response reuse. By grafting teacher-generated responses, BG-OPD provides additional guidance to further align the student with the teacher, yielding larger gains.

Effect of turn-selection strategy. We compare turn-selection strategies with $M = 8$ branches per trajectory and the same loss settings (Table 3). FTB Top-M (Chen et al., 2026) selects the M turns with the highest KL scores averaged over response tokens, Top M selects those with the highest turn scores s_t, and Random samples turns uniformly. BG-OPD samples turns with $P(t) \propto s_t$. Across both model scales, Top M outperforms FTB Top-M on average, and weighted sampling achieves the highest average task success rate. These results support the turn score as a useful signal for branch selection. They further favor weighted sampling, which prioritizes high-scoring turns while retaining opportunities for teacher guidance at other turns.

Table 3: **Effect of turn-selection strategy.**

Selection	Cybench	NYU CTF	XBOW	Avg
Qwen3.5-9B				
Random	15.00	11.06	21.15	15.74
FTB Top-M	12.50	11.56	17.12	13.73
Top M	15.00	13.07	23.08	17.05
Ours	22.50	14.57	25.96	**21.01**
Qwen3.5-4B				
Random	10.00	5.53	14.42	9.98
FTB Top-M	12.50	6.03	17.38	11.97
Top M	12.50	9.55	16.35	12.80
Ours	15.00	10.05	19.23	**14.76**

Hyperparameter sensitivity analysis. For both student sizes, Figure 5 varies M at $\lambda = 0.2$ and λ at $M = 8$. Increasing M from 1 to 8 improves performance by 7.87% and 7.07% on 9B and 4B, respectively, whereas $M = 16$ yields no additional gains and requires more teacher inference. Both models achieve their best performance at $\lambda = 0.2$, whereas increasing λ to 1 reduces performance by 7.07% and 7.73% from the peak, respectively. This decline suggests that a large branch loss weight may overemphasize supervision on single-turn teacher responses, reducing overall performance.

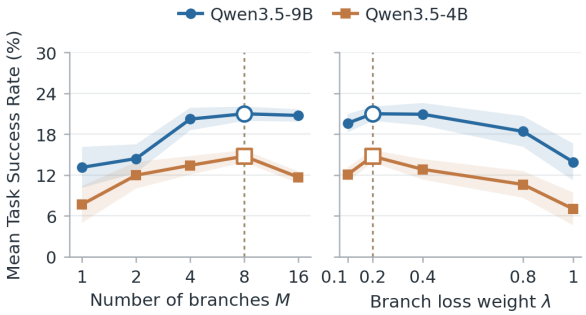


Figure 5: **BG-OPD hyperparameter sensitivity.** Number of teacher branches per trajectory M (left; $\lambda = 0.2$) and branch loss weight λ (right; $M = 8$).

Training dynamics. We analyze training dynamics from two perspectives: how student–environment interactions evolve over training, and how task performance and distillation signals change. **(1) Tool-use behavior during training.** Figure 6 reports tool and model calls per task and mean response length. After the initial updates, TurnOPD exhibits the lowest tool- and model-call counts. BG-OPD generally makes fewer calls per task than OPD and TCOD-F2B while generating

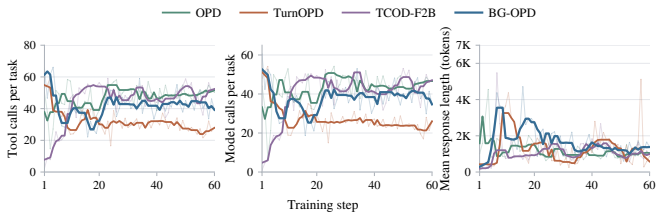
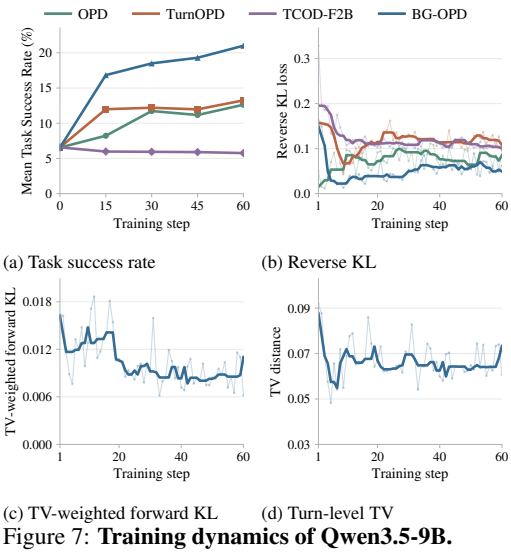


Figure 6: **Tool-use behavior during Qwen3.5-9B training.**



(a) Task success rate (b) Reverse KL

(c) TV-weighted forward KL (d) Turn-level TV

Figure 7: **Training dynamics of Qwen3.5-9B.**

longer responses. Combined with the task-success curves, these observations indicate that BG-OPD attains stronger performance than OPD and TCOD-F2B with fewer tool calls and longer responses per model call.

(2) Distillation signals during training. Figure 7 tracks mean task success rate and reverse KL for the four methods, together with BG-OPD's TV-weighted branch forward KL and mean turn-level TV. Reverse KL measures teacher–student mismatch on student-generated trajectories. The two BG-OPD signals measure mismatch on grafted teacher responses and disagreement at sampled turns, respectively. BG-OPD improves fastest and remains strongest after step 15. TurnOPD improves early and then plateaus, while TCOD-F2B remains nearly flat; both retain higher reverse KL than BG-OPD after their initial declines. Vanilla OPD reaches a lower reverse KL than these baselines but still trails BG-OPD in task success. Within BG-OPD, task success improves as both reverse KL and its TV-weighted branch forward KL decline. This joint trend suggests that forward-KL supervision on TV-selected grafted branches may further improve student–teacher alignment and task performance.

5 Related Work

CTF Agents and Training Environments. Recent work develops CTF agents through scaffold optimization and model training (Udeshi et al., 2025; Kong et al., 2025; Muzsai et al., 2025). One line of work improves agent scaffolds through self-evolution, as in CyberEvolver (Fan et al., 2026). Another fine-tunes models on pre-collected trajectories: Cyber-Zero (Zhuo et al., 2026) synthesizes trajectories from writeups, while CTF-Dojo (Zhuo et al., 2025) collects execution-verified trajectories in reconstructed environments. CTF agents must explore vulnerabilities and interact with running targets to obtain informative feedback for learning. This calls for scalable collections of realistic, executable CTF environments with verifiable flag recovery.

On-Policy Distillation for Agents. Recent agentic OPD methods mitigate compounding errors through rollout control and selective supervision. TCOD and TurnOPD regulate rollout depth and turn-level supervision (Wang et al., 2026; Zhou et al., 2026b). Guided-OPD (Li et al., 2026a) and DASH-OPD (Xia et al., 2026) mix teacher and student execution. SAGE-OPD (Zhou et al., 2026a) selectively weights teacher supervision. FutureBridge-OPD (Chen et al., 2026) evaluates teacher guidance using subsequent student outcomes. We graft teacher-generated response branches onto student trajectories to extend teacher guidance beyond student rollouts.

6 Conclusion

In this work, we systematically study how to transfer CTF capabilities from larger LLMs to smaller models through on-policy distillation. To address environment scarcity, we construct 1,370 verifiably solvable, containerized CTF environments through evidence-guided reconstruction and iterative verification. To extend teacher supervision beyond student-sampled trajectories, we propose BG-OPD, which samples turns according to teacher–student disagreement and grafts single-turn teacher-generated response branches onto student trajectories for additional distillation supervision, without executing branch actions. Experiments on three CTF benchmarks across two model scales demonstrate consistent improvements over vanilla OPD and other distillation baselines.

AI USE STATEMENT

Generative AI tools were used to generate initial manuscript drafts, polish the writing, and automate the batch submission of experimental jobs. The authors reviewed all AI-assisted content and experimental outputs and take full responsibility for the final manuscript and reported results.

ETHICS STATEMENT

Cybersecurity agents are inherently dual-use. Our goal is to advance automated security testing by training and evaluating agents in controlled CTF settings, but the resulting capabilities could be misused against systems without authorization. All experiments are conducted in isolated, containerized environments built from publicly released CTF artifacts or established benchmarks. These environments reproduce intentionally vulnerable challenges derived from public CTF artifacts. They do not include undisclosed vulnerabilities in deployed software. No public services or third-party systems are used as security targets. BG-OPD is a general training method. Models trained in this work will be made available to researchers only after rigorous capability and safety evaluation and release review.

REPRODUCIBILITY STATEMENT

The appendices provide additional experimental setting details (Appendix A.1), environment construction and validation details (Appendix A.2), and the CTF task interaction and flag-verification procedure (Appendix C).

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.

Anthropic. Claude Code overview, 2026. URL `https://code.claude.com/docs/en/overview`. Accessed: 2026-09-26.

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

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

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. `https://github.com/verl-project/uni-agent`, 2026. GitHub repository. Supervisor: Xibin Wu and Juntao Li.

Yihe Fan, Changyi Li, Lichen Xu, Xudong Pan, Jiarun Dai, Hong Geng, and Min Yang. Cyberevolver: Structured self-evolution for cybersecurity agents on the fly. *arXiv preprint arXiv:2605.26195*, 2026.

Zimo Ji, Daoyuan Wu, Wenyuan Jiang, Pingchuan Ma, Zongjie Li, and Shuai Wang. Measuring and augmenting large language models for solving capture-the-flag challenges. In *Proceedings of the 2025 ACM SIGSAC Conference on Computer and Communications Security*, pp. 603–617, 2025.

He Kong, Die Hu, Jingguo Ge, Liangxiong Li, Hui Li, and Tong Li. Pentest-r1: Towards autonomous penetration testing reasoning optimized via two-stage reinforcement learning. *arXiv preprint arXiv:2508.07382*, 2025.

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

Jiaze Li, Hao Yin, Haoran Xu, Boshen Xu, Wenhui Tan, Zewen He, Jianzhong Ju, Zhenbo Luo, and Jian Luan. Video-opd: Efficient post-training of multimodal large language models for temporal video grounding via on-policy distillation. *arXiv preprint arXiv:2602.02994*, 2026b.

Lajos Muzsai, David Imolai, and András Lukács. Improving llm agents with reinforcement learning on cryptographic ctf challenges. *arXiv preprint arXiv:2506.02048*, 2025.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026a. URL https://qwen.ai/blog?id=qwen3.5.

Qwen Team. Qwen3.6-27B: Flagship-level coding in a 27b dense model, April 2026b. URL https://qwen.ai/blog?id=qwen3.6-27b.

Minghao Shao, Sofija Jancheska, Meet Udeshi, Brendan Dolan-Gavitt, Haoran Xi, Kimberly Milner, Boyuan Chen, Max Yin, Siddharth Garg, Prashanth Krishnamurthy, et al. Nyu ctf bench: A scalable open-source benchmark dataset for evaluating llms in offensive security. *Advances in Neural Information Processing Systems*, 37:57472–57498, 2024.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. Hybridflow: A flexible and efficient rlhf framework. In *Proceedings of the Twentieth European Conference on Computer Systems*, EuroSys '25, pp. 1279–1297. ACM, March 2025. doi: 10.1145/3689031.3696075. URL http://dx.doi.org/10.1145/3689031.3696075.

Core Team, Bangjun Xiao, Bingquan Xia, Bo Yang, Bofei Gao, Bowen Shen, Chen Zhang, Chenhong He, Chiheng Lou, Fuli Luo, Gang Wang, Gang Xie, Hailin Zhang, Hanglong Lv, Hanyu Li, Heyu Chen, Hongshen Xu, Houbin Zhang, Huaqiu Liu, Jiangshan Duo, Jianyu Wei, Jiebao Xiao, Jinhao Dong, Jun Shi, Junhao Hu, Kainan Bao, Kang Zhou, Lei Li, Liang Zhao, Linghao Zhang, Peidian Li, Qianli Chen, Shaohui Liu, Shihua Yu, Shijie Cao, Shimao Chen, Shouqiu Yu, Shuo Liu, Tianling Zhou, Weijiang Su, Weikun Wang, Wenhan Ma, Xiangwei Deng, Bohan Mao, Bowen Ye, Can Cai, Chenghua Wang, Chengxuan Zhu, Chong Ma, Chun Chen, Chunan Li, Dawei Zhu, Deshan Xiao, Dong Zhang, Duo Zhang, Fangyue Liu, Feiyu Yang, Fengyuan Shi, Guoan Wang, Hao Tian, Hao Wu, Heng Qu, Hongfei Yi, Hongxu An, Hongyi Guan, Xing Zhang, Yifan Song, Yihan Yan, Yihao Zhao, Yingchun Lai, Yizhao Gao, Yu Cheng, Yuanyuan Tian, Yudong Wang, Zhen Tang, Zhengju Tang, Zhengtao Wen, Zhichao Song, Zhixian Zheng, Zihan Jiang, Jian Wen, Jiarui Sun, Jiawei Li, Jinlong Xue, Jun Xia, Kai Fang, Menghang Zhu, Nuo Chen, Qian Tu, Qihao Zhang, Qiying Wang, Rang Li, Rui Ma, Shaolei Zhang, Shengfan Wang, Shicheng Li, Shuhao Gu, Shuhuai Ren, Sirui Deng, Tao Guo, Tianyang Lu, Weiji Zhuang, Weikang Zhang, Weimin Xiong, Wenshan Huang, Wenyu Yang, Xin Zhang, Xing Yong, Xu Wang, Xueyang Xie, Yilin Jiang, Yixin Yang, Yongzhe He, Yu Tu, Yuanliang Dong, Yuchen Liu, Yue Ma, Yue Yu, Yuxing Xiang, Zhaojun Huang, Zhenru Lin, Zhipeng Xu, Zhiyang Chen, Zhonghua Deng, Zihan Zhang, and Zihao Yue. Mimo-v2-flash technical report, 2026. URL https://arxiv.org/abs/2601.02780.

Meet Udeshi, Minghao Shao, Haoran Xi, Nanda Rani, Kimberly Milner, Venkata Sai Charan Putrevu, Brendan Dolan-Gavitt, Sandeep Kumar Shukla, Prashanth Krishnamurthy, Farshad Khorrami, et al. D-cipher: Dynamic collaborative intelligent multi-agent system with planner and heterogeneous executors for offensive security. *arXiv preprint arXiv:2502.10931*, 2025.

Jiaqi Wang, Wenhao Zhang, Weijie Shi, Yaliang Li, and James Cheng. Tcod: Exploring temporal curriculum in on-policy distillation for multi-turn autonomous agents. *arXiv preprint arXiv:2604.24005*, 2026.

XBOW Engineering. XBOW validation benchmarks. https://github.com/xbow-engineering/validation-benchmarks, 2024. Accessed: 2026-09-24.

Yuchen Xia, Qianguo Sun, Chao Song, Junlong Wu, Yiyan Qi, and Yunjian Xu. Dash-opd: Discrepancy-aware switching with hysteresis for on-policy distillation. *arXiv preprint arXiv:2607.29078*, 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, 2025. URL https://arxiv.org/abs/2505.09388.

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

Hee Suk Yoon, Eunseop Yoon, Jaehyun Jang, SooHwan Eom, Ji Woo Hong, Mark Hasegawa-Johnson, Qi Dai, Chong Luo, and Chang D Yoo. Decomposed on-policy distillation for vision-language reasoning: Steering gradients for visual grounding. *arXiv preprint arXiv:2606.00564*, 2026.

Andy K Zhang, Neil Perry, Riya Dulepet, Joey Ji, Celeste Menders, Justin Lin, Eliot Jones, Gashon Hussein, Samantha Liu, Donovan Jasper, Pura Peetathawatchai, Ari Glenn, Vikram Sivashankar, Daniel Zamoshchin, Leo Glikbarg, Derek Askaryar, Haoxiang Yang, Aolin Zhang, Rishi Alluri, Nathan Tran, Rinnara Sangpisit, Kenny Oseleononmen, Dan Boneh, Daniel Ho, and Percy Liang. Cybench: A framework for evaluating cybersecurity capabilities and risks of language models. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 25094–25243, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/3e9412a9c1d93810ef3ef7825115016b-Paper-Conference.pdf.

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

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

Terry Yue Zhuo, Dingmin Wang, Hantian Ding, Varun Kumar, and Zijian Wang. Training language model agents to find vulnerabilities with CTF-Dojo, 2025. URL https://arxiv.org/abs/2508.18370.

Terry Yue Zhuo, Dingmin Wang, Hantian Ding, Varun Kumar, and Zijian Wang. Cyber-zero: Training cybersecurity agents without runtime. In *International Conference on Learning Representations*, volume 2026, pp. 154054–154184, 2026.

A Additional Experimental Settings

A.1 Additional Experimental Setting Details

Table 4: Detailed experimental settings.

Config	Vanilla OPD (4B / 9B)	TCOD-F2B (4B / 9B)	TurnOPD (4B / 9B)	BG-OPD (4B / 9B)
Optimization				
Optimizer	AdamW	AdamW	AdamW	AdamW
Learning rate	5×10^{-6}	5×10^{-6}	5×10^{-6}	5×10^{-6}
LR schedule	Constant	Constant	Constant	Constant
Warmup ratio	0.08	0.08	0.08	0.08
Gradient clipping	0.4	0.4	0.4	0.4
Training budget				
Epochs	1	1	1	1
Global batch size	16	16	16	16
Rollout batch size	1	1	1	1
Rollouts and distillation				
Max prompt tokens	16K	16K	16K	16K
Maximum context length	128K	128K	128K	128K
Max training turns	60	60	60	60
Rollouts per task n	1	1	1	1
Rollout temperature	0	0	0	0
Rollout top-p	0.95	0.95	0.95	0.95
Top-K	256	256	256	256
Method-specific				
Initial horizon	—	1	60	—
Minimum horizon	—	—	2	—
Updates per turn η	—	1	—	—
Probe interval	—	—	8	—
Probe warmup steps	—	—	3	—
Coverage quantile	—	—	0.8	—
Min. successful trajectories	—	—	4	—
EMA coefficient	—	—	0.3	—
Blend progress (s, e)	—	—	$(0, 1)$	—
Branches per trajectory M	—	—	—	8
Branch loss weight λ	—	—	—	0.2

We use verl (Sheng et al., 2025) and Uni-Agent (Ding et al., 2026) as the training frameworks for Offline SFT and the four OPD methods. All training rollouts use non-thinking mode.

For OPD Baselines and BG-OPD, detailed settings are provided in Table 4.

For Offline SFT, we use the teacher model (Qwen3.6-27B) to generate trajectories in our reconstructed CTF environments and retain 744 trajectories that successfully recover a valid flag through rejection sampling. We fine-tune each base student on this fixed dataset using AdamW with a learning rate of 1×10^{-5}, a warmup ratio of 0.05, a global batch size of 64, a maximum training context length of 128K tokens, and a weight decay of 0.01.

For Evaluation, all models are evaluated with the Claude Code scaffold (Anthropic, 2026) in non-thinking mode at temperature 0, using the default context-window limit and a maximum of 60 interaction turns per task. One task from NYU CTF Bench is excluded because its environment could not be reproduced.

A.2 Environment Construction Settings

We use DeepSeek-V4-Flash through the Claude Code agent interface to construct target environments and solver scripts. The pipeline selects a reconstruction route according to the available challenge materials: supplied Docker or Compose files are reused or repaired, while missing deployments are reconstructed from attachments, source code, and writeups when sufficient evidence is available. Solver scripts are adapted from existing exploits or generated from writeups and challenge artifacts. During construction, the agent invokes a separate programmatic verifier to build and deploy the target and execute the solver in isolation, with access restricted to player-visible attachments and target endpoints. Build errors, runtime errors, and flag-recovery results are returned to the agent for repair. Before retaining a candidate, we screen the player-visible materials for direct flag or complete-solution leakage and remove challenges that overlap with the evaluation benchmark to prevent contamination.

Table 5 summarizes our environment construction settings. Each task run allows up to 200 agent rounds (model calls) and eight verifier checks within a four-hour time limit. Each verifier check permits up to five solver execution attempts, stopping at the first valid flag recovery. These limits bound the construction, verification, and solver-retry loops, respectively. Each value is a configured budget for its loop.

Table 5: Environment construction settings. Round, attempt, and time values are upper limits.

Setting	**Value**
Construction model	DeepSeek-V4-Flash
Agent interface	Claude Code
Agent rounds per task run	200
Wall-clock time per task run	4 hours
Verifier checks per task run	8
Docker build timeout	30 minutes
Solver execution attempts per check	5 (stop on success)

B Residual Branching and Its Turn-Level Approximation

This appendix formalizes a response-level interpretation of residual weighting and relates it to the turn-level surrogate used by BG-OPD.

Response-level residual construction. At the fixed history h of Proposition 1, write $p(y) = \pi_{\theta_{\text{old}}}(y \mid h)$ and $q(y) = \pi_\phi(y \mid h)$. Both policies use a common response termination convention. Let $d^+(y) = [q(y) - p(y)]_+$ be the positive residual. Since p and q each sum to one, positive and negative differences have equal mass, so $\sum_y d^+(y) = \frac{1}{2}\sum_y |q(y) - p(y)| = \Delta$. The upper envelope $u = p + d^+ = \max(p, q)$ is the smallest measure dominating both distributions, with mass $1 + \Delta$. Its normalized target $\rho = u/(1 + \Delta)$ yields

$$(1 + \Delta) D_{\text{KL}}(\rho \| \pi_\theta(\cdot \mid h)) \equiv -\mathbb{E}_p[\log \pi_\theta(y \mid h)] - \sum_y d^+(y) \log \pi_\theta(y \mid h), \tag{12}$$

where $\equiv$ drops terms independent of θ. Under this response-level construction, the student distribution and positive residual define a single cross-entropy projection. The same construction applies to token distributions at a fixed prefix and to complete response distributions.

Residual sampling and ideal-target contraction. For $\Delta > 0$, draw $I \sim \text{Bernoulli}(\Delta)$ and, independently, $y \sim \nu = d^+/\Delta$. For any integrable loss f, $\mathbb{E}[If(y)] = \sum_y d^+(y) f(y)$; choosing $f(y) = -\log \pi_\theta(y \mid h)$ gives an unbiased residual estimator. Since $u \geq q$ pointwise,

$$[q(y) - \rho(y)]_+ \leq \frac{\Delta q(y)}{1 + \Delta} \quad \Longrightarrow \quad D_{\text{TV}}(q, \rho) \leq \frac{\Delta}{1 + \Delta} < \Delta. \tag{13}$$

For $\Delta = 0$, $p = q = \rho$ and the branch contribution is zero. Equation (13) characterizes contraction of the ideal target distribution.

Proof of Proposition 1. The exact residual factors as $d^+ = qw$, so $\mathbb{E}_q[w] = \Delta$. The ideal loss is therefore $\sum_y d^+(y)\ell(y) = \mathbb{E}_q[w\ell]$. Replacing $w(y)$ by s gives the turn-level proxy $s\mathbb{E}_q[\ell]$. Adding and subtracting $\Delta\mathbb{E}_q[\ell]$ yields

$$\mathbb{E}_q[w\ell] - s\mathbb{E}_q[\ell] = \mathbb{E}_q[(w-\Delta)\ell] + (\Delta - s)\mathbb{E}_q[\ell]. \tag{14}$$

The first term equals $\mathrm{Cov}_q(w, \ell)$, proving Equation (11). When $s = \Delta$, the approximation error equals $\mathrm{Cov}_q(w, \ell)$. The score s averages top-K token residuals along one student response. Each residual is a lower bound on the corresponding token-level TV. We interpret s as a turn-level surrogate whose error relative to response-level residual weighting is characterized exactly by Equation (11).

Relation to the implemented objective. Proposition 1 compares the same branch loss under different weights. Our branch loss uses token-averaged top-K forward KL at teacher prefixes. Equation (12) uses the response negative log-likelihood at the same prefixes. At a fixed prefix, forward KL and cross-entropy with the same teacher distribution have identical student gradients. The upper-envelope construction provides response-level motivation for the implemented branch loss. Its minimality and ideal-target contraction properties apply to the response-level objective in Equation (12). The full BG-OPD objective combines the reverse-KL OPD trunk loss, top-K forward-KL branch loss, and approximate residual weighting. Histories, scores, and generated responses are fixed during updates; gradients flow only through the student losses. Selection reuses the original teacher scores, while branches require teacher decoding and both models' probabilities at new prefixes. The main-text identities assume continuations sampled directly from π_ϕ. If decoding changes this distribution, the residual-loss comparison gains an additional term: s times the difference in expected branch loss between the teacher policy and the actual sampling distribution.

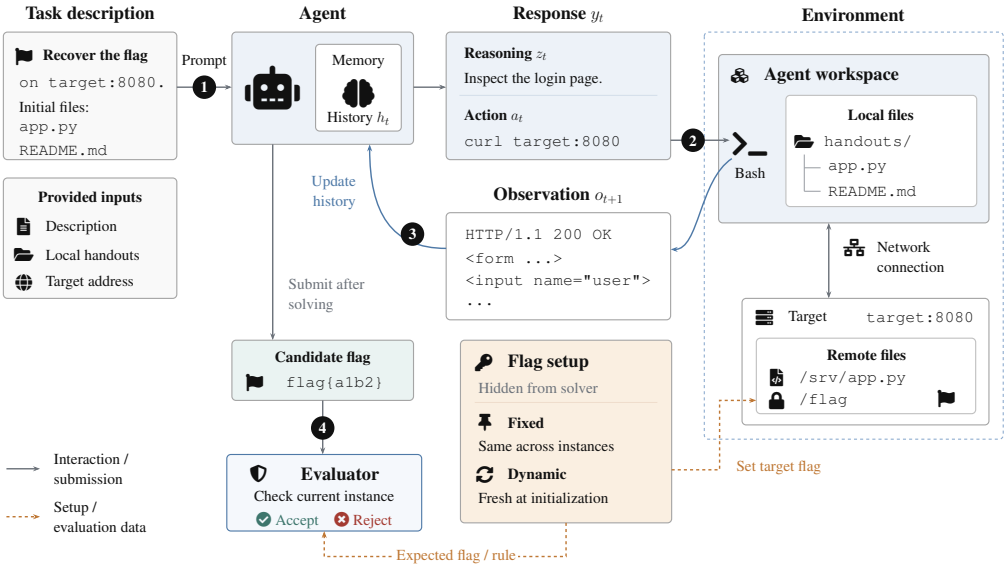
Unbiased estimation of the turn-weighted proxy. Condition on a rollout τ, its scores, and θ. For $W > 0$, $P(t) = s_t/W$ implies $(W/T)P(t) = s_t/T$. Iterated expectation and averaging over the M draws give

$$\mathbb{E}\left[\frac{W}{TM}\sum_{m=1}^{M}\ell(\theta; h_{J_m}, y_m^\phi)\,\middle|\,\tau\right] = \frac{1}{T}\sum_t s_t\mathbb{E}_{y\sim\pi_\phi(\cdot|h_t)}[\ell(\theta; h_t, y)]. \tag{15}$$

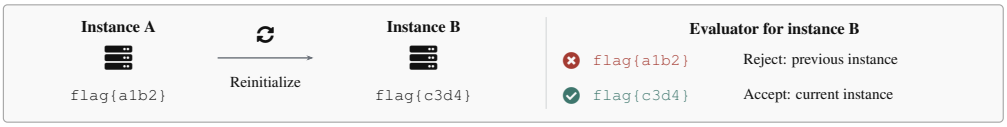
For $W = 0$, both contributions are zero. Thus the sampled contribution in Equation (8) is unbiased for the turn-weighted proxy. Increasing M reduces the sampling variance of this proxy. Proposition 1 characterizes its approximation error.

C CTF Task Interaction and Flag Verification

Figure 8 illustrates the CTF setup introduced in Section 2.1. The agent inspects local handouts and interacts with a target service through its workspace. File-only tasks omit the target service. Flag configuration is separate from the agent's inputs: the evaluator uses the expected flag or validation rule for the active instance, including when a fresh flag is generated at initialization.



(a) Service-based CTF interaction



(b) Dynamic flags across environment instances

Figure 8: CTF task interaction and flag verification, adapted from Figure 1 of Zhang et al. (2025). **(a)** The task description is passed to the agent (1), the response's action executes in the workspace (2), observations update the agent's history (3), and a submitted flag is checked by the evaluator (4). Local handouts and target-side files are shown separately. Dashed arrows carry setup and evaluation data unavailable directly to the agent. **(b)** With dynamic flags, reinitializing the same task produces a new expected value: a flag recovered from instance A is rejected in instance B. File names, command output, and flag strings are illustrative.