# WHAT GETS DEPLOYED? DEPLOYMENT ACCOUNTING FOR STATEFUL AGENTS

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Agentic RL studies usually report single-run success or oracle pass@$k$, yet practical systems may generate several sandboxed executions and release only one. These metrics can therefore misidentify both the best system and the reason it improves. We introduce *deployment accounting*: a protocol that jointly reports native success, successful commitment, wrong commitment, abstention, and oracle availability, and that decomposes policy differences into candidate quality and the selector's incremental value on identical pools. We instantiate a label-independent outcome selector and conduct a controlled 7B study on ALFWorld and a restricted WebShop setup. Selection adds roughly 9–13 success points to the same ALFWorld weights across the evaluated learned policies. Over four continuation seeds, a simple penalty-free outcome-RL control reaches 72.5% committed success in seen rooms and 71.3% in unseen rooms, compared with 62.9% and 60.5% for the original GRPO pipeline. A commit-aware variant reaches 79.2% in the designated unseen-room run and provides favorable low-error operating points, but a matched zero-credit control shows that its auxiliary credit is not required for the strongest average policy. Same-pool analyses attribute most average gains to better candidates rather than a larger voting increment; WebShop shows that fewer wrong commitments can instead come from more abstention. These results establish deployment accounting as a practical methodology for separating learning, selection, and coverage effects before attributing improvements in stateful-agent systems.

## 1 INTRODUCTION

Language-model agents change environments: they move objects, edit records, and purchase products. In a sandboxed system, several executions can be generated before one result is authorized for release. The relevant question is therefore not only whether an agent can succeed, but *what actually gets deployed*. A successful execution may be hidden among inconsistent alternatives, a frequent wrong outcome may win a vote, and an apparently safer system may simply decline more tasks.

Single-run accuracy measures candidate quality, while oracle pass@$k$ asks whether any candidate succeeds. Neither characterizes a label-independent release procedure. Execution-based consensus and large-scale sample filtering already exploit agreement among generated programs or answers (Li et al., 2022; Shi et al., 2022; Li et al., 2024; Wang et al., 2023); selective prediction studies the complementary decision to abstain (Geifman & El-Yaniv, 2017; 2019). Stateful agents combine both issues because different executions can leave different persistent effects.

We propose **deployment accounting**, a controlled methodology for separating three levers that are often conflated. **Learning** changes the distribution of candidate executions. **Selection** chooses among the realized effects. An **acceptance threshold** trades coverage for agreement. Deployment accounting evaluates all three under one commit contract, compares selectors on the same candidate pools, and uses matched learning controls before assigning gains to a particular credit rule.

Our empirical study includes ordinary group-relative RL, step-level and multi-sample objectives, a commit-aware credit hypothesis (CAPO), and a matched zero-credit control (D0). Every policy is evaluated with the same outcome-based selector on seen and unseen ALFWorld rooms; five WebShop seeds provide a boundary case in which error reduction and useful task completion move differently.



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.

**Contributions.** (1) We formalize a stateful commit-or-abstain protocol and an accompanying deployment-accounting report: native success, successful commitment, wrong commitment, abstention, oracle availability, and same-pool selector increments. (2) We provide a controlled learning study over four ALFWorld continuation seeds, five WebShop seeds, strong RL baselines, seen and unseen rooms, threshold curves, and matched reward-processing controls. (3) We establish a reusable empirical taxonomy of deployment gains. Outcome selection can improve fixed weights by roughly 10–12 points; learning can improve the candidate distribution without increasing the voter's incremental benefit; and lower wrong-commit rates can reflect either better accepted outcomes or lower coverage. (4) We show why attribution requires matched controls: the strongest average ALFWorld policy is a simple penalty-free outcome-RL control, while the more elaborate commit-aware variant remains competitive and offers favorable low-error operating points in a designated unseen-room run. The contribution is thus a methodology and set of controlled findings about deployed agent performance, not a claim that consensus certifies correctness or that one auxiliary credit is universally superior.

## 2 Deployment Accounting for Stateful Agents

For task $x$, let $\tau \sim \pi_\theta(\cdot \mid x)$ be an agent–environment execution with task success $r(\tau) \in \{0, 1\}$. A fixed abstraction $\sigma(\tau)$ records externally relevant effects. The designated class $\varnothing$ denotes no *tracked* effect; it is not a certificate that every environment variable is unchanged.

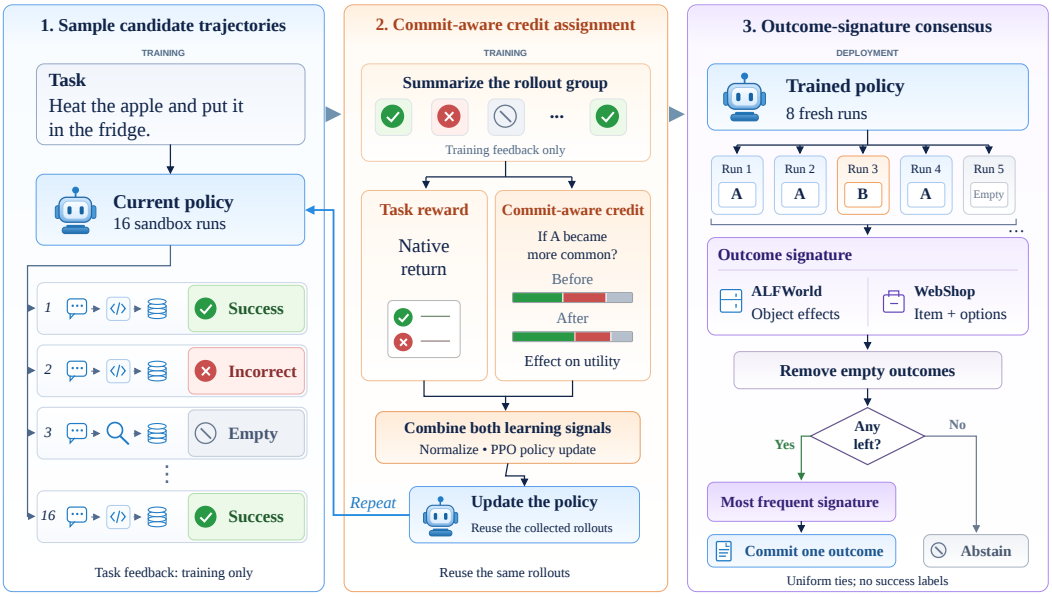


Figure 1: **Candidate learning and outcome-based deployment.** The center illustrates CAPO as one candidate-learning variant; all evaluated policies use the same deployment procedure on the right. Training may use task feedback, but deployment groups fresh sandbox executions by effect signatures, removes designated empty outcomes, and commits a modal nonempty result or abstains without correctness labels. Unselected effects are never released.

**Commit rule $D_{k,m}$.** Generate $k$ independent executions in isolated sandboxes. Remove empty outcomes, identify the largest remaining signature class, and commit only if its count is at least $m$. If none qualifies, abstain. Ties between largest classes and representatives within a selected class are uniform. The complete state of one execution is committed; states from different candidates are never merged. The primary operating point is $k = 8, m = 1$; threshold sweeps vary only $m$.

$$C_k = \mathbb{P}(\text{successful commit}), \qquad H_k = \mathbb{P}(\text{wrong commit}),$$
$$A_k = \mathbb{P}(\text{abstain}), \qquad C_k + H_k + A_k = 1. \tag{1}$$

Coverage is $1 - A_k$ and conditional error is $H_k/(C_k + H_k)$ when coverage is positive. Native success and oracle pass@$k$ describe generation; they are not substitutes for $C_k$.

**Effect representations.** ALFWorld signatures summarize tracked object locations, selected property changes, and toggles inferred from valid actions. WebShop signatures record the purchased item and observed options; no purchase maps to $\varnothing$. These maps are fixed across policies but are effect abstractions, not certified full-state equivalence classes. Appendix B documents known false splits and omissions.

Outcome plurality is empirical minimum-Bayes-risk selection under loss $\mathbf{1}[\sigma(\tau_i) \neq \sigma(\tau_j)]$, plus empty filtering and a reject option. It connects to execution-based selection in program synthesis and code generation (Li et al., 2022; Shi et al., 2022; Li et al., 2024). The stateful setting adds an explicit commitment boundary: no unselected candidate may release a payment, message, or physical action. This assumption is aligned with sandbox-based agent-risk evaluation (Ruan et al., 2024), but it does not make retrying irreversible live actions safe.

**Agreement is not verification.** Signatures contain no reward labels. A modal signature may be wrong, and a class can contain both successful and failed trajectories. We integrate the randomized representative choice and score the selected execution's actual label only after selection. Threshold curves are therefore selective-decision diagnostics, not calibrated probabilities or risk guarantees (Guo et al., 2017); conformal risk control would require a separate calibration set and assumptions not supplied by these experiments (Angelopoulos et al., 2024).

## 3 Candidate Policies Under a Common Deployment Contract

We hold the deployment rule fixed and vary how its candidates are learned. PPO and REINFORCE-style group baselines provide the common optimization backbone (Schulman et al., 2017; Shao et al., 2024; Ahmadian et al., 2024); method names below denote the stated adaptations, not full reproductions of every source paper.

Table 1: **Learning variants under the common deployment contract.** All methods are evaluated with the same $D_{8,1}$ rule.

| Variant | Learning signal and role |
|---|---|
| GRPO | Group-standardized task reward with the original invalid-action penalty. |
| GRPO 2×LR | Same baseline with doubled learning rate; tests a step-size explanation. |
| GiGPO | Episode- and repeated-state relative credit (Feng et al., 2025). |
| PKPO-style | Exact pass@8 subset credit, adapted from PKPO (Walder & Karkhanis, 2025). |
| MV-LOO | Whole-pool deployment utility minus leave-one-out utility, after Tang et al. (2025). |
| CAU | Fixed-$k$ subset-averaged leave-one-out credit for the same commit utility. |
| RH | Ternary reward: correct $+1$, empty $0$, nonempty error $-1$ (Mohamadi et al., 2025). |
| SA-RL | Earlier unnormalized selection-aware credit based on action-signature plurality. |
| CAPO | Native reward plus normalized plug-in outcome-frequency credit. |
| D0 ($\lambda = 0$) | Same direct CAPO code path with the auxiliary credit disabled; penalty-free outcome RL. |

**CAPO as a tested hypothesis.** Let $p_\theta(s) = \mathbb{P}_\theta(\sigma = s)$ and $q_\theta(s) = \mathbb{E}_\theta[r \mid \sigma = s]$. If $\alpha_s(p)$ is the probability that $D_k$ selects signature $s$, the deployment utility with wrong-commit cost $c$ is

$$\mathcal{U}_c(p, q) = \sum_{s \neq \varnothing} \alpha_s(p)[(1 + c)q_s - c] = C_k - cH_k. \tag{2}$$

CAPO estimates $p$ and within-class success from $M = 16$ rollouts, holds empirical $\hat{q}$ fixed, numerically differentiates $\mathcal{U}_c$ with respect to $\hat{p}$, and combines the centered sensitivity with standardized native reward. The main setting uses $c = 0$ and credit weight $\lambda = 1$. This reuses collected outcomes and adds no environment interactions.

The ideal signature-frequency gradient requires within-class correctness to remain fixed; the implemented plug-in credit, step-row whitening, and token-mean PPO reduction form a practical surrogate. Appendix A gives the identity, residual term, and finite-group computation. CAPO is therefore one experimental hypothesis inside a broader inference-aware RL lineage (Tang et al., 2025; Balashankar et al., 2025; Walder & Karkhanis, 2025), not a new general policy-gradient principle.

**The decisive matched control.** Direct ALFWorld CAPO omits the baseline's $-0.1$ invalid-action penalty. D0 preserves the same checkpoint, optimizer, update count, reward path, normalization, and code, changing only $\lambda$ from 1 to 0. Numerical contract checks identify D0 with penalty-free GRPO to within $9 \times 10^{-6}$ in recorded variable-length batches. Thus CAPO–D0 isolates the added outcome-frequency credit, while D0–GRPO captures the combined reward-processing difference.

# 4 Experimental Design

The common model is Qwen2.5-7B-Instruct, trained with full-parameter FSDP in `verl-agent` and the veRL/vLLM stack (Qwen et al., 2024; Feng et al., 2025; Sheng et al., 2025; Kwon et al., 2023). The study first holds deployment fixed while varying learning, then holds candidate pools fixed while varying the selector.

**ALFWorld.** Seeds 17, 29, 43, and 53 continue from the same 30-update GRPO warm start for 30 additional updates. Each update contains eight tasks with 16 executions per task; the horizon is 50 actions. These are continuation seeds, not independently trained warm starts. Evaluation schedules 128 positions per generation run. Reconstructing the recorded environment schedule yields 91 distinct seen-room games and 74 unseen-room games, with no gamefile overlap between splits. Historical rollout logs retain aligned initial observations rather than gamefile IDs; Appendix B distinguishes reconstruction from direct logging.

**WebShop.** The restricted setup uses a 1,000-product index, synthetic goals, a 15-action horizon, and 128 fixed evaluation positions. Training and evaluation goal IDs are disjoint. Five CAPO/GRPO seeds are evaluated at update 10; GiGPO and RH are available for seed 17. This is a controlled low-success setting, not a protocol-matched WebShop leaderboard experiment.

**Comparability and published baselines.** The ALFWorld setup is a short continuation study from one warm start, with a different prompt, task schedule, training horizon, and deployment metric from published GiGPO results. We therefore compare methods only within this controlled setup and do not treat external headline scores as directly comparable. All main continuations share task sampling, group size, update budget, context construction, and downstream selector. The variants differ in the declared reward/credit paths summarized in Table 1.

**Evaluation and uncertainty.** The final ALFWorld tables use the first eight generation runs at temperature 0.4. Ties are integrated label-independently. Older sixteen-run analyses form two disjoint @8 pools and are reported separately; overlapping eight- and sixteen-run analyses are not independent replications. WebShop uses sixteen runs. We report every completed training seed. Seed standard deviations describe continuation variability; nominal paired seed-$t$ intervals and task-cluster bootstrap intervals answer different questions (Henderson et al., 2018; Agarwal et al., 2021). No best checkpoint or favorable seed subset replaces the fixed last checkpoint.

**Integrity and scope.** All compared runs use aligned evaluation positions, and the analysis records no parsing or reward omissions. Exact task manifests, reward calculations, normalization order, signature parsers, and checkpoint identifiers are part of the reproducibility contract because this paper's conclusions depend on small implementation differences. The appendix provides the recorded settings and known representation limitations; the main results do not claim full-benchmark SOTA.

# 5 What Changes Deployment Performance?

## 5.1 Selection changes the result even before learning

For a fixed Qwen2.5-7B policy on ALFWorld, native success is 19.2%, action-set plurality at $k = 5$ is 19.7%, and empty-filtered outcome selection reaches 29.9% (oracle pass@5: 43.0%). For the common GRPO-30 start, the corresponding values are 33.1%, 34.5%, and 44.8% at $k = 6$ (oracle: 52.3%). These are within-policy comparisons with additional inference: they show that the release rule matters, not that weights improved.

### 5.2 LEARNING AND SELECTION ARE COMPLEMENTARY

Table 2: **Four ALFWorld continuation seeds, common first-eight-run evaluation.** Mean percentages. Native@1 evaluates one attempt; $C_8/H_8/A_8$ evaluate the same policy under outcome-based deployment. Seed standard deviations and per-seed values are in Appendix C.

| Split | Learned policy | Native@1 | $C_8$ | $H_8$ | $A_8$ |
|---|---|---|---|---|---|
| Seen | GRPO | 53.6 | 62.9 | 33.4 | 3.7 |
| Seen | CAU | 53.5 | 64.5 | 29.0 | 6.4 |
| Seen | CAPO | 55.5 | 66.0 | 29.1 | 4.9 |
| Seen | D0 ($\lambda = 0$) | 61.8 | 72.5 | 24.2 | 3.3 |
| Unseen | GRPO | 50.5 | 60.5 | 32.3 | 7.2 |
| Unseen | CAU | 51.7 | 63.2 | 30.5 | 6.2 |
| Unseen | CAPO | 55.4 | 68.5 | 24.3 | 7.2 |
| Unseen | D0 ($\lambda = 0$) | 59.3 | 71.3 | 24.0 | 4.7 |

The strongest average candidate policy is D0, the simple penalty-free outcome-RL control. It reaches $72.5\%$ committed success in seen rooms and $71.3\%$ in unseen rooms, compared with GRPO's $62.9\%$ and $60.5\%$. Deployment adds value to every learned policy in Table 2: $C_8 - \mathrm{Native@1}$ ranges from 9.3 to 11.0 points in seen rooms and from 10.0 to 13.1 points in unseen rooms. For D0 specifically, the same weights improve by 10.7 and 12.0 points. Thus candidate learning and release-time selection are complementary empirical effects rather than competing explanations. D0 is numerically above GRPO in every continuation seed in both splits, although paired seed-level intervals remain wide ($[-2.6, +21.7]$ and $[-2.2, +23.8]$ points).

CAPO is also a strong observed policy: its mean unseen-room $C_8$ is 68.5%, and CAPO–GRPO unseen differences are positive in all four seeds ($+16.5, +5.4, +3.5, +6.5$ points; mean $+8.0 \pm 5.8$). The interval $[-1.3, +17.3]$ retains uncertainty about another continuation. These results establish useful trained policies and transfer to unseen rooms; attribution to a particular auxiliary credit is a separate question.

Table 3: **Designated seed-17 comparison at the same $k = 8$ budget.** A compact set is shown here; the complete ten-method table is in Appendix C. Bold denotes the best point estimate, not established dominance.

| | Seen rooms | | Unseen rooms | |
|---|---|---|---|---|
| Method | $C_8 \uparrow$ | $H_8 \downarrow$ | $C_8 \uparrow$ | $H_8 \downarrow$ |
| GRPO | 60.2 | 34.3 | 62.7 | 33.4 |
| PKPO-style | 60.0 | 29.8 | 68.9 | 27.2 |
| RH | 53.9 | 27.3 | 48.1 | 33.9 |
| GiGPO | 71.5 | 27.0 | 65.1 | 33.3 |
| CAU | 74.3 | 22.5 | 72.7 | 24.2 |
| D0 ($\lambda = 0$) | **79.7** | **17.9** | 76.8 | 18.5 |
| CAPO | 78.8 | 19.6 | **79.2** | **16.1** |

## 6 POLICY AND THRESHOLD ARE DISTINCT CONTROL KNOBS

A frozen policy is not a single deployment point. Holding its $k = 8$ candidate pool fixed, increasing the minimum modal count $m$ trades coverage for agreement. Figure 2 plots every evaluated threshold rather than selecting one favorable point.

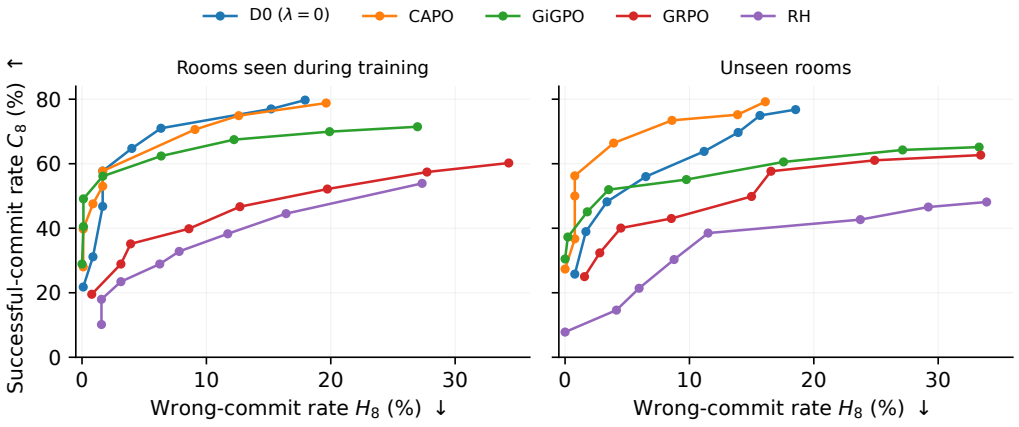


Figure 2: **Successful-commit rate $C_8$ versus wrong-commit rate $H_8$.** Seed 17, first-eight-run pools, $m = 1, \ldots, 8$. Lines connect observed operating points; they are not confidence bands or calibrated frontiers. The default ranking changes with the operating threshold, especially on unseen rooms.

D0 has the strongest default seen-room point. On unseen rooms, however, CAPO offers favorable low-error operating points: at the common threshold $m = 4$, CAPO yields $C_8/H_8 = 66.4/3.9\%$ and D0 yields $63.8/11.2\%$; at $m = 5$, they yield $56.2/0.8\%$ and $56.0/6.5\%$. These comparisons are descriptive and post-hoc—equal $m$ does not imply equal coverage, and seed 17 is not a multi-seed frontier estimate—but they show why a default success ranking is not a complete deployment decision.

More attempts can amplify both successes and errors. In the earlier sixteen-run seed-17 analysis, increasing $k$ from 1 to 16 raises CAPO's $C_k$ from 68.0% to 79.1% and GRPO's from 51.1% to 62.0%. Yet GRPO's $H_k$ also rises from 27.6% to 36.4%, while CAPO's changes from 18.8% to 19.3%. Extra candidates can reinforce an incorrect mode, so pass@$k$ and committed success must be reported with wrong commitments.

The policy and threshold are therefore complementary: training determines the candidate distribution; $m$ controls which realized patterns are accepted. Agreement is not calibrated confidence. A deployment requiring a risk guarantee would need a separate validation protocol, for example conformal risk control under its assumptions (Angelopoulos et al., 2024), rather than choosing a threshold on evaluation labels.

# 7 A Controlled Decomposition of the Gains

## 7.1 Candidate quality versus selector increment

Evaluate every selector on the same candidate pool and the same empty mask. Let $C_R$ be success when choosing uniformly among nonempty candidates, assigning zero to all-empty pools, and let $S = C_D - C_R$ be the increment from plurality. Then $\Delta C_D = \Delta C_R + \Delta S$. This is an accounting identity, not causal mediation.

Table 4: **Same candidates and coverage, different selectors.** Mean success (%) over four ALFWorld continuation seeds, first-eight-run pools.

| Split | Policy | Random nonempty | Action plurality | Outcome plurality |
|---|---|---|---|---|
| Seen | GRPO | 59.4 | 63.6 | 62.9 |
| Seen | D0 | 68.3 | 71.5 | 72.5 |
| Seen | CAU | 60.6 | 63.7 | 64.5 |
| Seen | CAPO | 63.5 | 67.2 | 66.0 |
| Unseen | GRPO | 55.4 | 60.6 | 60.5 |
| Unseen | D0 | 65.2 | 70.2 | 71.3 |
| Unseen | CAU | 59.3 | 62.6 | 63.2 |
| Unseen | CAPO | 62.4 | 68.5 | 68.5 |

D0 and CAPO chiefly improve the candidate-composition term relative to GRPO. Outcome plurality improves over uniform nonempty selection, but does not consistently exceed action plurality. CAPO's mean additional plurality increment over GRPO is $+1.0$ point in the eight-run unseen analysis, $+0.7$ in the sixteen-run unseen supplement, and $-0.6$ in the older seen analysis. Thus the main average gain is better generated outcomes, not a reliably larger voting increment. This is a substantive learned-policy improvement, but a different mechanism from making the voter itself more effective.

### 7.2 A MATCHED CONTROL CHANGES THE CAUSAL STORY

Table 5: **Matched attribution over four continuation seeds.** Percentage-point differences with nominal paired seed-$t$ 95% intervals ($n = 4$).

| Contrast | Measure | Seen rooms | Unseen rooms |
|---|---|---|---|
| D0 – GRPO | $\Delta C_8$ | $+9.6\,[-2.6, +21.7]$ | $+10.8\,[-2.2, +23.8]$ |
| CAPO – GRPO | $\Delta C_8$ | $+3.1\,[-13.3, +19.5]$ | $+8.0\,[-1.3, +17.3]$ |
| CAPO – D0 | $\Delta C_8$ | $-6.5\,[-14.6, +1.6]$ | $-2.8\,[-16.1, +10.5]$ |
| CAPO – D0 | $\Delta H_8$ | $+4.9\,[+0.1, +9.8]$ | $+0.3\,[-9.9, +10.5]$ |
| CAPO – CAU | $\Delta C_8$ | $+1.5\,[-3.5, +6.5]$ | $+5.3\,[-3.6, +14.1]$ |
| CAPO – CAU | $\Delta H_8$ | $+0.1\,[-6.2, +6.4]$ | $-6.2\,[-11.8, -0.7]$ |

D0 retains CAPO's direct code path but disables the auxiliary credit. It exceeds CAPO in seen-room committed success in every seed, averaging 6.5 points, and has 4.9 points fewer wrong commitments. CAPO and CAU, which estimate the same deployment utility differently, have unresolved success differences. The controlled conclusion is therefore not that the deployment question is unimportant; it is that the extra outcome-frequency derivative is not required for the strongest average policy in this setup.

Reward processing is a plausible alternative explanation. In an all-failure group where fraction $f$ of rows receives penalty $-\varepsilon$ and the rest zero, population standardization gives

$$z_{\text{penalized}} = -\sqrt{(1-f)/f}, \qquad z_{\text{other}} = \sqrt{f/(1-f)}. \tag{3}$$

The nominal penalty magnitude cancels. A rare small invalid-action penalty can therefore create a substantial relative learning signal when all task rewards fail. Logged invalid-row frequencies are 0.15–0.9%. This analytic diagnostic and the D0 control expose reward-processing sensitivity, but do not prove that penalty amplification caused the performance difference; a factorial penalty-by-normalization intervention was not completed. The general lesson is methodological: in sparse, group-normalized agent RL, matched reward-processing controls are necessary before attributing a gain to a more elaborate credit rule (Engstrom et al., 2020; Liu et al., 2025; Yu et al., 2025).

# 8 A BOUNDARY CASE AND RELATION TO PRIOR WORK

## 8.1 WEBSHOP: FEWER WRONG PURCHASES CAN MEAN MORE ABSTENTION

Across five seeds in the restricted WebShop setup, GRPO has mean $C_8/H_8/A_8 = 17.61/41.46/40.94\%$ and CAPO has $16.54/33.23/50.23\%$. Wrong commitments fall by $8.23 \pm 6.90$ points, but abstention rises by $9.30 \pm 5.12$ and committed success changes by $-1.07 \pm 4.29$. At matched coverage, random thinning gives $+1.47 \pm 3.72$ points and vote-threshold matching gives $+1.22 \pm 3.94$. Thus the robust observation is a coverage–error trade-off, not a general success gain. In seed 17, CAPO is more productive than RH but less conservative; its matched-coverage advantage over RH is uncertain. This boundary case demonstrates why lower unconditional error cannot be equated with better accepted-result quality.

## 8.2 RELATION TO EXECUTION CONSENSUS AND INFERENCE-AWARE LEARNING

Large-scale sampling, filtering, and behavioral clustering have long improved code generation, notably in AlphaCode (Li et al., 2022). Self-consistency aggregates sampled answers (Wang et al., 2023; Chen et al., 2023); MBR-EXEC and DOCE use execution behavior as a semantic proxy (Shi et al., 2022; Li et al., 2024). Tang et al. optimize multi-sample inference objectives, InfAlign aligns training with inference, and PKPO optimizes pass@$k$ (Tang et al., 2025; Balashankar et al., 2025; Walder & Karkhanis, 2025). We do not claim a new general voting or score-function principle. Our contribution is a controlled empirical study of learned candidate policies, effect-based selection, empty filtering, and abstention under a stateful commitment boundary.

ReAct and Reflexion exemplify reasoning-and-feedback agent loops (Yao et al., 2023; Shinn et al., 2023). Tool-using benchmarks and agent training expose persistent environmental consequences (Shridhar et al., 2021; Yao et al., 2022; Zhou et al., 2024; Liu et al., 2024; Trivedi et al., 2024; Chen et al., 2025; Feng et al., 2025). ToolEmu studies risk through an LM-emulated sandbox (Ruan et al., 2024); our work instead evaluates actual sandbox effects in two benchmarks and assumes release only after selection. Selective classification and reject-option learning study risk jointly with coverage (Geifman & El-Yaniv, 2017; 2019; Mohamadi et al., 2025). Our agreement score is not calibrated probability; conformal risk control provides a distinct route to finite-sample risk guarantees when a suitable calibration set and monotone loss are available (Angelopoulos et al., 2024).

**Scope.** The ALFWorld runs are short continuations from one warm start, not independent end-to-end trainings. WebShop is a restricted low-success configuration. Signatures can omit relevant changes or split semantically equivalent outcomes, and the full training artifacts are needed for end-to-end replication. These limits narrow the claim to the performed study: deployment-aware metrics and matched controls materially change how its agent policies should be interpreted.

# 9 DISCUSSION

The study supports a stronger conclusion than "run more samples," but a narrower one than universal algorithmic dominance. Three distinct mechanisms can change the released result. First, training changes the candidate distribution, as seen most clearly for D0 on ALFWorld. Second, selection converts a fixed pool into a better returned result, adding roughly 9–13 points across the evaluated ALFWorld policies. Third, an agreement threshold reduces wrong commitments by lowering coverage and can change which policy is preferable. Reporting only native accuracy, pass@$k$, or wrong-commit rate conflates these mechanisms.

The matched controls also change the scientific interpretation. CAPO produces a high-performing designated policy and favorable low-error unseen-room operating points, but the zero-credit control is the stronger default on average. This does not erase the learned-policy or deployment results; it shows that reward processing can matter more than a theoretically motivated auxiliary term. Likewise, the same-pool decomposition shows that learned candidate quality, rather than a special increase in plurality's incremental benefit, explains most average gains. The central empirical lesson is therefore about *deployment attribution*: a useful system outcome and the mechanism claimed to produce it are separate propositions.

### 9.1 A Minimum Deployment-Accounting Protocol

The analyses suggest a compact protocol for future stateful-agent studies. **First**, report native success, oracle pass@$k$, successful commitment, wrong commitment, abstention, and compute under one fixed budget. **Second**, compare selectors on identical candidate pools and identical empty masks; otherwise candidate generation and selection cannot be separated. **Third**, report threshold or risk–coverage curves rather than one favorable operating point, and choose production thresholds on held-out calibration data. **Fourth**, distinguish training seeds from repeated evaluation generations and preserve cluster structure in uncertainty estimates. **Fifth**, use a matched reward-processing control before attributing a gain to an auxiliary credit. These requirements are lightweight relative to agent training, but they materially changed every headline interpretation in this study.

**Scope and open problems.** The outcome representation remains an important limitation. The current signatures are fixed and transparent, but they are not canonical final-state equivalence. Future work should compare action traces, tracked effects, and canonical simulator-state deltas on an independently defined audit set. The threshold curves are descriptive rather than calibrated; deployment at a specified risk requires validation-based or conformal calibration. Finally, the ALFWorld seeds share one warm start, WebShop is a restricted low-success setting, and protocol-complete benchmark runs and independent warm starts are needed to estimate algorithm-level generalization.

## 10 Conclusion

A stateful agent's useful output is the result it releases, not merely an attempted trajectory or an oracle-correct member of a set. We formalized deployment accounting for a commit-or-abstain system and used it to study reinforcement-learned candidate policies under a common inference budget. The experiments reveal concrete gains: outcome selection improves fixed and learned policies, and learned candidates can reach substantially higher committed success on both seen and unseen ALFWorld rooms. The strongest average policy in this study is a simple penalty-free outcome-RL control, while a commit-aware variant offers distinct low-error operating points in a designated unseen-room run.

More importantly, the study identifies why superficially similar improvements can mean different things. Better candidate distributions can increase deployment success without increasing the voter's incremental benefit. Lower wrong-commit rates can result from abstention rather than more accurate accepted decisions. An auxiliary credit can produce a strong policy without being necessary for the strongest average result. By evaluating these effects separately—and by retaining unfavorable controls in the main paper—deployment accounting prevents a useful system outcome from being assigned to the wrong mechanism.

The findings also change how test-time scaling should be interpreted for agents. More candidates expose more opportunities for success, but they can also reinforce a recurring wrong outcome. A release procedure must therefore be evaluated jointly with the policy that supplies its candidates, the threshold that controls coverage, and the cost assigned to an incorrect state change. The same frozen policy can support different operating points; the policy with the best default success need not be the policy with the best low-error regime.

The cross-environment comparison makes this distinction concrete. On ALFWorld, learned candidate quality supports higher observed commitment success; on WebShop, fewer wrong purchases mainly accompany more abstention. Neither pattern should be reduced to a single accuracy number. Retaining both results, along with the matched control and all continuation seeds, makes the accounting protocol informative even when a proposed credit does not improve the strongest baseline.

The practical recommendation is straightforward. Evaluate learning and release together; compare selectors on the same candidates; report success, error, abstention, oracle availability, and compute; and isolate reward processing before claiming improved credit assignment. Outcome-based deployment is not a correctness oracle or a substitute for safer environments. It is a disciplined experimental methodology for deciding which generated effects reach the world and for understanding what actually made an agent system better.

## AI USE STATEMENT

Generative AI tools were used to assist with manuscript writing, editing, and translation. The authors take responsibility for the final content.

## REPRODUCIBILITY STATEMENT

The appendices specify the deployment contract, training variants, evaluation pools, and complete available seed results. The aggregate inputs and scripts used to regenerate the tables and arithmetic checks will be released upon publication. Full checkpoints, original rollout records, task manifests, and launch environments are required for an end-to-end rerun and should accompany an archival release. We distinguish recorded observations from reconstructed manifests throughout.

## ETHICS STATEMENT

The evaluated commitment rule assumes isolated sandbox executions whose external effects are released only after a decision. It must not be interpreted as a mechanism for undoing already-issued payments, messages, physical actions, or privacy-sensitive operations. Wrong-commit rates in these benchmarks measure task-level incorrect state changes, not the severity of real-world harm.

## REFERENCES

Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron Courville, and Marc G. Bellemare. Deep Reinforcement Learning at the Edge of the Statistical Precipice. In *Advances in Neural Information Processing Systems*, 2021.

Arash Ahmadian, Chris Cremer, Matthias Gallé, Marzieh Fadaee, Julia Kreutzer, Olivier Pietquin, Ahmet Üstün, and Sara Hooker. Back to Basics: Revisiting REINFORCE-Style Optimization for Learning from Human Feedback in LLMs. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12248–12267, 2024. doi: 10.18653/v1/2024.acl-long.662.

Anastasios N. Angelopoulos, Stephen Bates, Adam Fisch, Lihua Lei, and Tal Schuster. Conformal Risk Control. In *International Conference on Learning Representations*, 2024.

Ananth Balashankar, Ziteng Sun, Jonathan Berant, Jacob Eisenstein, Michael Collins, Adrian Hutter, Jong Lee, Chirag Nagpal, Flavien Prost, Aradhana Sinha, Ananda Theertha Suresh, and Ahmad Beirami. InfAlign: Inference-aware language model alignment. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267, pp. 2646–2672, 2025.

Kevin Chen, Marco Cusumano-Towner, Brody Huval, Aleksei Petrenko, Jackson Hamburger, Vladlen Koltun, and Philipp Krähenbühl. Reinforcement Learning for Long-Horizon Interactive LLM Agents. *arXiv preprint arXiv:2502.01600*, 2025.

Xinyun Chen, Renat Aksitov, Uri Alon, Jie Ren, Kefan Xiao, Pengcheng Yin, Sushant Prakash, Charles Sutton, Xuezhi Wang, and Denny Zhou. Universal Self-Consistency for Large Language Model Generation. *arXiv preprint arXiv:2311.17311*, 2023.

Logan Engstrom, Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Firdaus Janoos, Larry Rudolph, and Aleksander Madry. Implementation Matters in Deep Policy Gradients: A Case Study on PPO and TRPO. In *International Conference on Learning Representations*, 2020.

Lang Feng, Zhenghai Xue, Tingcong Liu, and Bo An. Group-in-Group Policy Optimization for LLM Agent Training. In *Advances in Neural Information Processing Systems*, 2025.

Yonatan Geifman and Ran El-Yaniv. Selective Classification for Deep Neural Networks. In *Advances in Neural Information Processing Systems*, pp. 4878–4887, 2017.

Yonatan Geifman and Ran El-Yaniv. SelectiveNet: A Deep Neural Network with an Integrated Reject Option. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97, pp. 2151–2159, 2019.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On Calibration of Modern Neural Networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70, pp. 1321–1330, 2017.

Peter Henderson, Riashat Islam, Philip Bachman, Joelle Pineau, Doina Precup, and David Meger. Deep Reinforcement Learning That Matters. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, pp. 3207–3214, 2018. doi: 10.1609/aaai.v32i1.11694.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient Memory Management for Large Language Model Serving with PagedAttention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, 2023. doi: 10.1145/3600006.3613165.

Haau-Sing Li, Patrick Fernandes, Iryna Gurevych, and André F. T. Martins. DOCE: Finding the Sweet Spot for Execution-Based Code Generation. *arXiv preprint arXiv:2408.13745*, 2024.

Yujia Li, David Choi, Junyoung Chung, Nate Kushman, Julian Schrittwieser, Rémi Leblond, Tom Eccles, James Keeling, Felix Gimeno, Agustin Dal Lago, et al. Competition-Level Code Generation with AlphaCode. *Science*, 378(6624):1092–1097, 2022. doi: 10.1126/science.abq1158.

Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, et al. AgentBench: Evaluating LLMs as Agents. In *International Conference on Learning Representations*, 2024.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding R1-Zero-Like Training: A Critical Perspective. In *Conference on Language Modeling*, 2025.

Mohamad Amin Mohamadi, Tianhao Wang, and Zhiyuan Li. Honesty over Accuracy: Trustworthy Language Models through Reinforced Hesitation. *arXiv preprint arXiv:2511.11500*, 2025.

Qwen, An Yang, Baosong Yang, et al. Qwen2.5 Technical Report. *arXiv preprint arXiv:2412.15115*, 2024.

Yangjun Ruan, Honghua Dong, Andrew Wang, Silviu Pitis, Yongchao Zhou, Jimmy Ba, Yann Dubois, Chris J. Maddison, and Tatsunori Hashimoto. Identifying the Risks of LM Agents with an LM-Emulated Sandbox. In *International Conference on Learning Representations*, 2024.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal Policy Optimization Algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models. *arXiv preprint arXiv:2402.03300*, 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*, 2025.

Freda Shi, Daniel Fried, Marjan Ghazvininejad, Luke Zettlemoyer, and Sida I. Wang. Natural Language to Code Translation with Execution. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 3533–3546, 2022. doi: 10.18653/v1/2022.emnlp-main.231.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language Agents with Verbal Reinforcement Learning. In *Advances in Neural Information Processing Systems*, 2023.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. ALFWorld: Aligning Text and Embodied Environments for Interactive Learning. In *International Conference on Learning Representations*, 2021.

Yunhao Tang, Kunhao Zheng, Gabriel Synnaeve, and Remi Munos. Optimizing Language Models for Inference Time Objectives using Reinforcement Learning. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267, pp. 59066–59085, 2025.

Harsh Trivedi, Tushar Khot, Mareike Hartmann, Ruskin Manku, Vinty Dong, Edward Li, Shashank Gupta, Ashish Sabharwal, and Niranjan Balasubramanian. AppWorld: A Controllable World of Apps and People for Benchmarking Interactive Coding Agents. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 16022–16076, 2024. doi: 10.18653/v1/2024.acl-long.850.

Christian Walder and Deep Tejas Karkhanis. Pass@K Policy Optimization: Solving Harder Reinforcement Learning Problems. In *Advances in Neural Information Processing Systems*, 2025. doi: 10.52202/085713-5096.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-Consistency Improves Chain of Thought Reasoning in Language Models. In *International Conference on Learning Representations*, 2023.

Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8:229–256, 1992. doi: 10.1007/BF00992696.

Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. WebShop: Towards Scalable Real-World Web Interaction with Grounded Language Agents. In *Advances in Neural Information Processing Systems*, 2022.

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.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, et al. DAPO: An Open-Source LLM Reinforcement Learning System at Scale. In *Advances in Neural Information Processing Systems*, 2025.

Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. WebArena: A Realistic Web Environment for Building Autonomous Agents. In *International Conference on Learning Representations*, 2024.

# A Utility Derivatives and Implementation Scope

## A.1 Proof of the ideal credit identity

**Proposition 1** (Fixed within-class correctness)**.** *For a fixed finite support, a fixed selector, and $q_\theta(s) = q(s)$ independent of $\theta$, define $g_s = \partial \mathcal{U}_c / \partial p_s$. Under score-function regularity,*

$$\nabla_\theta \mathcal{U}_c = \mathbb{E}\left[\left(g_{\sigma(\tau)} - \sum_s p_s g_s\right) \nabla_\theta \log \pi_\theta(\tau)\right]. \tag{4}$$

A reward-sufficient signature is a sufficient condition; empirical purity is not a proof. This is a specialization of standard score-function and inference-objective reasoning (Williams, 1992; Tang et al., 2025). Under these assumptions,

$$\nabla_\theta \mathcal{U}_c(p_\theta, q) = \sum_s g_s \nabla_\theta p_\theta(s) \tag{5}$$

$$= \sum_s g_s \mathbb{E}[\mathbf{1}\{\sigma(\tau) = s\} \nabla_\theta \log \pi_\theta(\tau)] \tag{6}$$

$$= \mathbb{E}[g_{\sigma(\tau)} \nabla_\theta \log \pi_\theta(\tau)]. \tag{7}$$

Subtracting any outcome-independent baseline preserves this identity; choosing $\sum_s p_s g_s$ centers the credit. These are standard score-function manipulations. They require that the differentiated policy distribution and the data-generating distribution are correctly identified and that the relevant expectations can be differentiated.

For a general abstraction, let $\alpha_s(p)$ be the probability that the deployment chooses signature $s$. Then

$$\mathcal{U}_c(p, q) = \sum_{s \neq \varnothing} \alpha_s(p)[(1 + c)q_s - c]. \tag{8}$$

If $q = q_\theta$, the full derivative also contains

$$\sum_s \frac{\partial \mathcal{U}_c}{\partial q_s} \nabla_\theta q_s = \mathbb{E}\left[\frac{(1 + c)\alpha_\sigma(p)}{p_\sigma}(r - q_\sigma) \nabla_\theta \log \pi_\theta(\tau)\right] \tag{9}$$

for positive-probability nonempty classes. This term accounts for changes in correctness *inside* a class. A simple counterexample has one nonempty signature shared by both successful and failed trajectories: $p_s = 1$ is fixed while $q_\theta(s)$ changes. Frequency credit alone cannot recover the whole utility gradient. The performed CAPO experiments do not add this residual; they hold empirical $\hat{q}$ fixed when computing the auxiliary signal.

## A.2 Finite-group computation

For observed classes, the implementation evaluates multinomial count configurations for deployment budget $k$ under the empirical class probabilities. For count vector $n$ and the tied modal nonempty classes $T(n)$, define $u_c(n; q) = 0$ when no nonempty class exists and otherwise

$$u_c(n; q) = \frac{1}{|T(n)|} \sum_{s \in T(n)} [(1 + c)q_s - c], \qquad \mathcal{U}_c = \sum_{\sum_s n_s = k} \frac{k!}{\prod_s n_s!} \prod_s p_s^{n_s} \, u_c(n; q). \tag{10}$$

This integrates uniform class ties and representatives without selecting by correctness. Finite differences perturb one empirical probability by $10^{-3}$, clip, renormalize, and use the same empirical within-class correctness. The count fallback samples 20,000 multinomial vectors when exact enumeration would exceed 60,000 configurations. A common random seed across perturbations reduces numerical variation; it is not an additional correctness signal.

The implementation computes centered numerical sensitivities,

$$\widetilde{g}_s = \frac{\mathcal{U}_c(\mathrm{norm}(\hat{p} + \epsilon e_s), \hat{q}) - \mathcal{U}_c(\mathrm{norm}(\hat{p} - \epsilon e_s), \hat{q})}{2\epsilon}, \qquad \hat{g}_s = \widetilde{g}_s - \sum_v \hat{p}_v \widetilde{g}_v. \tag{11}$$

Clipping and normalization of perturbed probabilities, plus the empirical fixed $\hat{q}$, determine this practical computation. Let $z(v) = (v - \bar{v})/(\mathrm{sd}(v) + \delta)$ with the implementation's small variance floor. A constant vector returns zero. After forming $b_i$, step-row duplication weights a trajectory according to its number of recorded rows in the subsequent normalization. The final reduction is token-mean. Consequently, the neural-network surrogate is not exactly the ideal complete-trajectory score-function expectation. All claims about practical performance refer to the performed surrogate, not to an unimplemented estimator.

### A.3 Error costs and abstention

Since $C_k + H_k + A_k = 1$ and $A_k = p_{\varnothing}^k$ under independent sampling,

$$U_c = (1 + c)U_0 - c + c\,p_{\varnothing}^k. \tag{12}$$

Thus $c$ can affect even groups containing observed successes if the empty-outcome term is relevant. Its effect is not restricted to all-failure groups. At $c = 0$ with no observed success, the empirical utility is zero everywhere and the extra credit is inactive. Neither case implies that the learned policy will be safe in a new environment.

### A.4 Penalty normalization as a diagnostic

For all-failure rows with reward $-\varepsilon$ on fraction $f$ and zero otherwise, the mean is $-f\varepsilon$ and population standard deviation is $\varepsilon\sqrt{f(1-f)}$. Equation 3 follows immediately. With the variance floor or a sample-SD convention the exact magnitude changes. The calculation demonstrates an induced relative advantage for unpenalized failures; it does not prove that their absolute policy probability, much less deployment error, necessarily increases after a coupled neural-network update. A factorial penalty-by-normalization training intervention would be needed to identify the proposed mechanism. It was not part of the completed study.

## B Algorithm and Reproducibility Details

**Algorithm 1** One practical CAPO collection/update cycle

**Require:** Current policy, task batch, frozen outcome abstraction; $M = 16, k = 8, c = 0, \lambda = 1$

**for** each task **do**
  Collect $M$ complete executions in separate environment instances
  Store generated actions, step rows, task rewards, and label-independent signatures
  Estimate empirical class frequencies $\hat{p}$ and within-class success $\hat{q}$
  Evaluate the fixed deployment utility over counts (exact or declared fallback)
  Compute centered finite-difference credit with $\hat{q}$ fixed
  Combine standardized native reward and credit; broadcast and whiten step rows
  Use the direct advantage or the declared baseline-preserving correction
**end for**
Apply one PPO epoch with the recorded KL, entropy, clipping, and token reduction
Record update count, reward processing, activity statistics, and checkpoint identity

Table 6: Training settings recorded in the completed campaign. No unrecorded optimizer defaults are inferred. D0/CAU launch arguments were compared with the corresponding CAPO/CAU runs.

| Setting | Value |
|---|---|
| Main model / update scope | Qwen2.5-7B-Instruct / full parameters, FSDP |
| Rollouts per update | 8 tasks $\times$ 16 executions |
| Learning rate | $10^{-6}$, constant |
| PPO epochs / minibatch | 1 / 128 |
| Microbatch per GPU | 8 |
| PPO clip / dual clip | 0.2 / 3.0 |
| KL / entropy coefficients | 0.01 / 0.001 |
| Loss reduction | Token mean; group step-row advantage whitening |
| Training decoding | Temperature 1; top-$p = 1$; at most 512 response tokens per step |
| Agent context | Task, two-step history, available actions, think/action format |
| Direct ALFWorld penalty | Omitted for CAPO, D0, and the declared direct controls |
| WebShop correction | Baseline GRPO advantage plus CAPO-minus-binary-task-reward correction |
| Memory | CPU parameter/optimizer offload, gradient checkpointing, padding removal |
| Hardware / software | 4 H100 80GB per run; Python 3.12; PyTorch 2.8.0+cu128; vLLM 0.11.0; Transformers 4.57.3 |
| Framework | `verl-agent`; Ray 2.58; FlashAttention 2.8.3 |

**Baseline implementation summary.** GRPO uses group-standardized task reward and the original invalid-action penalty. GRPO $2\times$LR changes only the learning rate. SA-RL adds an unnormalized action-plurality sensitivity to the GRPO advantage. MV-LOO uses the full-pool commit utility minus the utility after removing one trajectory. CAU averages fixed-$k$ subset utilities and uses a leave-one-out baseline for the same deployment objective. PKPO-style uses exact pass@8 subset credit. RH standardizes a ternary execution reward (correct, empty, nonempty error). All direct ALFWorld controls use the same single PPO epoch and token-mean reduction; method-specific penalty handling is declared in Table 1. In WebShop, the baseline-preserving correction subtracts the standardized binary task-reward advantage from the helper credit before adding it to the GRPO advantage.

**Starts, checkpoints, and protocol differences.** ALFWorld uses a single GRPO-30 warm start followed by 30 updates for all matched continuations. The maximum horizon is 50. Success returns reward 10; native dense scores follow the recorded clipping path. WebShop uses the original instruct checkpoint, at most 15 actions, and exact task success for the binary outcome. The early 15-update runs are evaluated at update 10 because the final artifacts were incomplete for both methods; later runs were scheduled for ten updates. The selected checkpoint was shared across compared methods, but constant learning rate alone does not establish equality of every scheduling detail. Complete optimizer state and exact model revision remain part of the original run configuration, not inferred values in this manuscript.

**Outcome abstractions and representation errors.** ALFWorld's map reconstructs tracked object effects from valid actions. For example, taking an object and replacing it on its original surface can leave a location entry even though the true net location is unchanged. Opening a receptacle is ignored. WebShop records item identity and observed option selections; accumulating option values can distinguish a red-then-blue path from a blue-only path even when the last selection overwrites the same option category. These limitations are properties of the performed representation. We do not silently correct evaluation signatures after examining results or claim that observed high purity proves exact semantic equivalence.

**Game reconstruction and uncertainty.** The completed integrity check rebuilt evaluation scheduling with worker seeds 1000–1127. The 128 ID positions map to 91 distinct games from the seen-room split; 128 OOD positions map to 74 distinct games from the unseen-room split. OOD initial-observation hashes merge some distinct games, yielding 54 groups, while ID has 91. Original generation records contain aligned initial observations but not the corresponding gamefile field.

Reconstruction and observed alignment are reported as such. Reported task-bootstrap intervals use the original initial-observation groups, not a fabricated count of unique rooms.

The eight-run follow-up was specified before its new outcomes were examined, but after the earlier CAPO results motivated the controls. Its primary intervals are paired seed-level $t$ intervals over four continuations. Existing two-level bootstrap intervals are supplementary and can differ, especially with four seeds. The two disjoint pools of the older 16-run analysis share policies and evaluation positions; their estimates are not extra training replications. We report nominal intervals without interpreting the number of significant per-seed comparisons as a test of the pooled effect.

# C Complete Available Results

Table 7: All completed main ALFWorld policies, common first-eight-run evaluation. Percentages; these rows are the source of the main-paper all-seed means.

| Split | Method/seed | Native | $C_8$ | $H_8$ | $A_8$ | Oracle | $C_R$ |
|---|---|---|---|---|---|---|---|
| Seen | GRPO/17 | 51.2 | 60.2 | 34.3 | 5.5 | 74.2 | 58.6 |
| Seen | GRPO/29 | 49.4 | 61.1 | 37.3 | 1.6 | 73.4 | 54.6 |
| Seen | GRPO/43 | 54.5 | 61.7 | 33.6 | 4.7 | 79.7 | 60.0 |
| Seen | GRPO/53 | 59.4 | 68.7 | 28.2 | 3.1 | 76.6 | 64.3 |
| Seen | D0 ($\lambda = 0$)/17 | 67.5 | 79.7 | 17.9 | 2.3 | 90.6 | 74.9 |
| Seen | D0 ($\lambda = 0$)/29 | 60.3 | 72.6 | 25.8 | 1.6 | 81.2 | 65.9 |
| Seen | D0 ($\lambda = 0$)/43 | 54.8 | 66.7 | 27.1 | 6.2 | 75.8 | 63.2 |
| Seen | D0 ($\lambda = 0$)/53 | 64.7 | 71.1 | 25.8 | 3.1 | 82.0 | 69.0 |
| Seen | CAU/17 | 63.4 | 74.3 | 22.5 | 3.1 | 83.6 | 68.8 |
| Seen | CAU/29 | 49.2 | 61.8 | 31.2 | 7.0 | 68.8 | 57.9 |
| Seen | CAU/43 | 47.3 | 59.6 | 30.3 | 10.2 | 68.0 | 56.9 |
| Seen | CAU/53 | 54.1 | 62.5 | 32.0 | 5.5 | 73.4 | 59.1 |
| Seen | CAPO/17 | 68.2 | 78.8 | 19.6 | 1.6 | 86.7 | 74.3 |
| Seen | CAPO/29 | 46.7 | 59.6 | 31.8 | 8.6 | 71.9 | 57.1 |
| Seen | CAPO/43 | 51.4 | 59.6 | 35.7 | 4.7 | 76.6 | 59.5 |
| Seen | CAPO/53 | 55.8 | 66.2 | 29.1 | 4.7 | 78.1 | 63.0 |
| Unseen | GRPO/17 | 53.8 | 62.7 | 33.4 | 3.9 | 76.6 | 58.0 |
| Unseen | GRPO/29 | 42.3 | 54.8 | 41.3 | 3.9 | 67.2 | 46.7 |
| Unseen | GRPO/43 | 55.1 | 61.0 | 30.4 | 8.6 | 73.4 | 58.8 |
| Unseen | GRPO/53 | 50.7 | 63.5 | 24.0 | 12.5 | 73.4 | 58.1 |
| Unseen | D0 ($\lambda = 0$)/17 | 63.6 | 76.8 | 18.5 | 4.7 | 81.2 | 70.6 |
| Unseen | D0 ($\lambda = 0$)/29 | 61.7 | 75.4 | 21.5 | 3.1 | 82.8 | 67.8 |
| Unseen | D0 ($\lambda = 0$)/43 | 50.7 | 63.5 | 30.2 | 6.2 | 73.4 | 57.3 |
| Unseen | D0 ($\lambda = 0$)/53 | 61.2 | 69.5 | 25.8 | 4.7 | 82.0 | 65.1 |
| Unseen | CAU/17 | 62.0 | 72.7 | 24.2 | 3.1 | 88.3 | 67.3 |
| Unseen | CAU/29 | 46.8 | 56.3 | 36.7 | 7.0 | 71.1 | 55.0 |
| Unseen | CAU/43 | 52.1 | 66.0 | 27.8 | 6.2 | 75.8 | 61.4 |
| Unseen | CAU/53 | 46.0 | 58.0 | 33.4 | 8.6 | 78.1 | 53.4 |
| Unseen | CAPO/17 | 69.4 | 79.2 | 16.1 | 4.7 | 90.6 | 74.1 |
| Unseen | CAPO/29 | 46.3 | 60.1 | 31.3 | 8.6 | 72.7 | 54.5 |
| Unseen | CAPO/43 | 50.9 | 64.6 | 26.0 | 9.4 | 75.0 | 58.4 |
| Unseen | CAPO/53 | 55.1 | 70.0 | 23.7 | 6.2 | 83.6 | 62.5 |

Table 8: All five completed WebShop CAPO/GRPO seed pairs and the available GiGPO/RH seed-17 baselines. Sixteen generation runs form two disjoint @8 pools. All methods share the restricted 1,000-product synthetic-goal configuration.

| Method | Seed | Native | $C_8$ | $H_8$ | $A_8$ | $C_R$ |
|---|---|---|---|---|---|---|
| GRPO | 17 | 5.7 | 14.0 | 43.1 | 43.0 | 14.4 |
| GRPO | 29 | 6.6 | 17.4 | 36.5 | 46.1 | 17.4 |
| GRPO | 43 | 6.6 | 8.8 | 61.5 | 29.7 | 10.5 |
| GRPO | 53 | 10.2 | 26.5 | 33.3 | 40.2 | 24.8 |
| GRPO | 61 | 6.6 | 21.3 | 32.9 | 45.7 | 20.8 |
| CAPO | 17 | 5.8 | 16.8 | 29.7 | 53.5 | 15.8 |
| CAPO | 29 | 3.5 | 11.5 | 26.8 | 61.7 | 11.1 |
| CAPO | 43 | 4.9 | 12.9 | 46.1 | 41.0 | 11.7 |
| CAPO | 53 | 8.7 | 23.2 | 34.6 | 42.2 | 20.9 |
| CAPO | 61 | 6.9 | 18.3 | 29.0 | 52.7 | 17.8 |
| GiGPO | 17 | 7.2 | 15.6 | 43.8 | 40.6 | 15.7 |
| RH | 17 | 0.9 | 4.2 | 10.2 | 85.5 | 4.2 |

### C.1 Primary eight-run and supplementary sixteen-run results

The matched D0/CAU study uses first-eight-run policy scores throughout; the main tables do not compare a D0 eight-run score with a CAPO sixteen-run score. The earlier ALFWorld seed-17 figures of 79.0% ID and 79.4% OOD committed success use 16 runs; the corresponding eight-run values are 78.8% and 79.2%. Across four CAPO/GRPO continuation seeds, the older ID sixteen-run difference is $+4.13 \pm 9.46$ points, with the recorded two-level interval $[-3.39, +13.78]$. OOD gives $+8.00 \pm 5.82$ with eight runs and $+7.29 \pm 7.06$ with sixteen. OOD wrong-commit changes are negative in four seeds at eight runs but in only three at sixteen; seed 53 changes to $+0.67$ points in the latter. No result is promoted because the finite pool is more favorable.

### C.2 Matched coverage and selected-outcome quality

For a policy with coverage $v = C + H$, random thinning to $v' \leq v$ yields $C' = C(v'/v)$ and $H' = H(v'/v)$. This is a label-independent reference curve; it does not represent an independently calibrated deployment policy. A stronger curve requires a fixed vote-count threshold, allowing randomization only between prespecified neighboring operating points if necessary. Curves chosen using evaluation aggregates are reported as post-hoc diagnostics.

In ALFWorld ID, CAPO minus D0 at matched coverage is $-5.31 \pm 3.21$ points with seed-$t$ interval $[-10.42, -0.20]$. In OOD it is $-0.93 \pm 6.75$, interval $[-11.67, +9.81]$. CAPO minus CAU at OOD matched coverage is about $+5.9 \pm 4.1$, with an interval containing zero. These results constrain any claim that lower wrong-commit frequency by itself establishes a better policy.

## D Additional Diagnostics and Boundary Results

Table 9: Earlier ALFWorld seed-17 budget curve, 16-run data. These are not common-first-eight-run comparisons against D0.

| Method/metric | $k = 1$ | 2 | 4 | 8 | 16 |
|---|---|---|---|---|---|
| CAPO $C_k$ | 68.0 | 70.8 | 75.6 | 79.0 | 79.1 |
| GRPO $C_k$ | 51.1 | 54.2 | 57.8 | 61.1 | 62.0 |
| CAPO $H_k$ | 18.8 | 21.0 | 20.5 | 19.4 | 19.3 |
| GRPO $H_k$ | 27.6 | 31.4 | 32.8 | 34.6 | 36.4 |

**Sensitivity is substantial.** The primary $c = 0, \lambda = 1$ setting was not replaced by a post-hoc best hyperparameter. The recorded $\lambda = 2$ ALFWorld ablation substantially reduced performance, and $c = 1$ also lowered committed success in the tested setting. Adding the CAPO correction to GiGPO yielded 65.6% committed success, below the standalone GiGPO value of 71.3% in the corresponding

16-run seed-17 comparison. These observations argue against treating the credit as a universally beneficial drop-in component.

**Other model and environment configurations.** Qwen3-4B-Instruct-2507, trained directly from instruct for 30 ALFWorld updates, did not reproduce the 7B warm-started result: seed 17 committed success was 39.4% versus GRPO 40.5%, and seed 29 was 32.8% versus 40.8%. This comparison changes both model and starting-competence regime, so it does not isolate architecture as the cause. Mistral and Llama WebShop trials likewise did not establish a general success improvement; very low success and differing nonempty rates restrict their interpretation. An AppWorld Qwen3.5-9B LoRA boundary trial reported OCS@8 77.0% versus RLOO 76.5%, with interval for the difference spanning zero, while native success was lower. Sparse empty outcomes there are a possible descriptive difference, not a proven explanation for the null result.

**Finite-model checks.** The recorded enumeration-based tests compared plug-in and leave-one-out estimates on finite outcome distributions. For the selected settings, plug-in relative MSE was 0.63 at $M = 9$ and 0.37 at $M = 16$, versus 4.86 and 0.90 for the reference leave-one-out estimator; active-group fractions were 97%/100% versus 27%/87%. These are measurements under those constructed distributions. Numerical ordering checks found no violation of one frequent-wrong penalty ordering in 5,419 tested configurations, but another empty-outcome ordering had a counterexample at $c = 0$. We report these as diagnostic observations, not distribution-free theorems or proof that the practical credit is a Hájek projection.

**Compute.** The completed D0/CAU extension consumed 182.1 GPU-hours: 137.1 in training, 41.5 in new evaluation, and 3.5 for the existing CAU seed-17 OOD checkpoint. The broader record reports approximately 24 GPU-hours for a 30-update ALFWorld run and 6–8 GPU-hours for sixteen evaluations, versus approximately 4 and 3–4 GPU-hours in the short WebShop setting; these are approximate run costs, not a reconciled total campaign ledger. The matched tables include only completed runs under the stated contracts. Exploratory campaigns with different models, environments, or objectives are excluded from these comparisons.

**Reproducibility scope.** Aggregate files and table-regeneration scripts will be released upon publication, together with the original per-trajectory records and training states where feasible. A complete independent rerun requires those artifacts and their original environment configuration. In deployment, rejected sandbox candidates must not release effects before commitment. A wrong-commit reduction in these benchmarks is not evidence of safety for irreversible financial, physical, or privacy-sensitive actions.

## E D0-INCLUSIVE SUCCESS–ERROR CURVES

This supplementary diagnostic varies only the consensus threshold of the fixed $k = 8$ deployment: commit when the modal nonempty signature has at least $m$ votes, for $m = 1, \ldots, 8$, and otherwise abstain. It does not retrain policies or change the training cost $c$. Each split uses training seed 17 and the same 128 evaluation positions, with the first eight generation runs per method.

Figure 2 in the main paper shows the same operating points with corrected metric labels.

Table 10: **All plotted CAPO and D0 operating points.** Percentages computed from the stored aggregate curves; $H_8 = \text{coverage} - C_8$. The plot provides no pointwise confidence intervals. Using the same threshold does not force equal coverage or equal wrong-commit rate.

| | Seen rooms | | | | Unseen rooms | | | |
|---|---|---|---|---|---|---|---|---|
| | CAPO | | D0 | | CAPO | | D0 | |
| $m$ | $C_8$ | $H_8$ | $C_8$ | $H_8$ | $C_8$ | $H_8$ | $C_8$ | $H_8$ |
| 1 | 78.8 | 19.6 | 79.7 | 17.9 | 79.2 | 16.1 | 76.8 | 18.5 |
| 2 | 74.9 | 12.6 | 77.0 | 15.2 | 75.2 | 13.9 | 75.0 | 15.7 |
| 3 | 70.6 | 9.1 | 71.0 | 6.3 | 73.4 | 8.6 | 69.7 | 13.9 |
| 4 | 57.7 | 1.7 | 64.7 | 4.0 | 66.4 | 3.9 | 63.8 | 11.2 |
| 5 | 53.0 | 1.7 | 57.7 | 1.7 | 56.2 | 0.8 | 56.0 | 6.5 |
| 6 | 47.6 | 0.9 | 46.8 | 1.7 | 50.0 | 0.8 | 48.2 | 3.4 |
| 7 | 39.7 | 0.1 | 31.2 | 0.9 | 36.7 | 0.8 | 39.0 | 1.7 |
| 8 | 28.0 | 0.1 | 21.8 | 0.1 | 27.3 | 0.0 | 25.8 | 0.8 |

**Interpretation.** In seen rooms, CAPO and D0 exchange advantages across operating points; D0 has higher default committed success. In unseen rooms, several CAPO points offer more success with fewer wrong commitments than the corresponding D0 points: at $m = 4$, $C_8/H_8$ is $66.4/3.9$ for CAPO versus $63.8/11.2$ for D0. This is a descriptive example chosen after examining the curves, not a newly selected primary operating point. The plot qualifies any inference from the default threshold alone, but does not overturn the four-seed attribution results in Table 5. No statistical dominance or risk guarantee is inferred from connected point estimates.

# F Additional Context: Learning and Selection Beyond the Main Setting

This section separates an earlier AppWorld learning comparison from the matched ALFWorld/WebShop study. It is not an extra seed or an additional controlled environment for the same training implementation. The source record uses Qwen3.5-9B LoRA, an AppWorld-specific outcome map, the official 57-task development set, and sixteen generation runs. The archived policy labels identify checkpoint 20 for the learned variants.

Table 11: **Exploratory AppWorld comparison, same reported evaluation protocol within this table.** Percentages; the policy initially used a rules scaffold. These data are not pooled with the main Qwen2.5-7B full-parameter study or with later frozen-model outcome-consistency selection (OCS) scores.

| Policy | Native@1 | OCS@8 | Oracle pass@8 |
|---|---|---|---|
| Rules-only start | 57.8 | 65.7 | 82.5 |
| Outcome RL (binary task reward; RLOO) | 73.4 | 76.5 | 86.8 |
| SA | 72.5 | 80.4 | 86.0 |
| CAPO | 69.1 | 77.0 | 87.7 |

**What the comparison supports.** The completed archive reports OCS@8 increases over its rules-only start of 10.81 points for outcome RL (task-paired bootstrap interval $[+2.58, +19.91]$), 14.75 for SA ($[+6.03, +23.76]$), and 11.33 for CAPO ($[+3.29, +20.06]$). These are positive learning observations in this separate setting. The CAPO–outcome-RL difference is only 0.52 points ($[-6.27, +7.98]$), while CAPO's native success is 4.28 points lower ($[-8.11, -0.33]$). The result supports studying trained candidates and outcome selection together, not a unique advantage of CAPO credit.

The earlier analysis declared a 15-task contested subset, but the table above reports all 57 tasks rather than using a favorable subset as the headline. The archival output does not establish a new test-set SOTA claim, an independent final test, or exact parity with the main training contract. A very small empty-outcome rate was also reported, but is not experimentally isolated as the reason for the limited CAPO–outcome-RL difference.

## F.1 Concrete Boundary Results in Additional Model Configurations

Table 12: **Additional configurations from the completed record.** Success and wrong-commit rates; each comparison is local to its row pair. These are not a pooled multi-model mean.

| Environment/model | Seed | CAPO | | GRPO | |
|---|---|---|---|---|---|
| | | $C_8$ | $H_8$ | $C_8$ | $H_8$ |
| ALFWorld / Qwen3-4B | 17 | 39.4 | 37.9 | 40.5 | 38.8 |
| ALFWorld / Qwen3-4B | 29 | 32.8 | 47.3 | 40.8 | 37.3 |
| WebShop / Mistral-7B | 17 | 2.9 | 66.6 | 0.4 | 8.6 |
| WebShop / Llama-3.1-8B | 17 | 0.0 | 2.3 | 0.0 | 0.0 |

Mistral illustrates why success alone is insufficient: a small increase accompanies substantially more wrong commitments as abstention falls. Llama's eight evaluated runs have no successful candidate; empty rejection cannot manufacture one. The later Llama repeats were not run after observing zero success, and this outcome-dependent stop is disclosed. These boundaries complement the positive learned policies rather than being silently excluded from a general claim.