STATE-GROUNDED CREDIT ASSIGNMENT FOR DUAL-CONTROL TOOL AGENTS: SEPARATING PROGRESS FROM ATTRIBUTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Multi-turn tool agents in dual-control environments complete tasks jointly with users who can also act on a shared world. Standard GRPO with terminal rewards evaluates the resulting joint trajectory but does not distinguish progress produced directly by the agent from progress made through user-side operations. We introduce state-grounded process rewards that measure progress toward target states constructed from reference actions, assigning direct credit to assistant-side changes and factual delegated credit to user-side changes. We further introduce delegated credit estimation (DCE), which adjusts user-side credit by replaying user turns after replacing the assistant's current message with a neutral acknowledgement. Across three τ^2-bench-Verified domains, Qwen3.5-9B jointly trained with direct and factual delegated supervision achieves higher mean scores than Terminal GRPO and an adapted tool-call-matching baseline across all reported metrics. On Telecom, the online variant of DCE reaches 84.8% Pass^5 (success in all five trials), exceeding Terminal GRPO by 9.1 percentage points and factual delegated supervision by 2.1 points, with comparable single-trial success to the latter. Validation with separate replays shows that online DCE selectively reduces credit for user-side progress that recurs under the neutral message. These results support user-side state supervision and contextual refinement of delegated credit.

1 INTRODUCTION

Dual-control environments provide the assistant and the user with different tools for modifying a shared environment state. The assistant executes actions directly and instructs the user to carry out others. Task outcomes therefore reflect state changes produced by both participants. However, binary terminal rewards assign the same failure signal to all unsuccessful trajectories, regardless of their progress. Under group-relative policy optimization (GRPO) (Shao et al., 2024), trajectories with identical terminal rewards receive identical terminal advantages within a task group. In an all-failure group, terminal feedback therefore provides no preference between partial completion and no progress. Intermediate state changes can provide process feedback that distinguishes trajectories with the same terminal reward.

Tool-call matching can supplement terminal feedback by scoring predicted tool names and arguments against reference calls (Qian et al., 2025; Abdelaziz et al., 2026). State-based supervision evaluates the resulting state changes. The distinction is useful when different action sequences can produce the same target state. In dual-control tasks, state-based supervision must also account for progress produced by user-side operations.

We propose *role-decomposed state-grounded process rewards*. These rewards measure progress toward target states obtained by executing each task's reference actions, while allowing the policy to use a different action sequence. Tracking assistant- and user-controlled fields separately yields *direct progress* and *factual delegated progress*, respectively. Reaching a target value yields positive progress, while leaving it yields negative progress. These signals complement terminal GRPO with feedback on partial completion, including changes that would be absent from assistant-only state tracking.

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.

User-side progress need not depend on the current assistant instruction, since the user may already have the information needed to act. *Delegated credit estimation* (DCE) uses neutral-message user-turn replays to construct fixed field-level weights or contextual discounts for factual delegated progress. We evaluate the resulting policies and examine whether DCE reduces credit for progress that also occurs under a neutral message.

We evaluate Qwen3.5-9B policies jointly trained across three τ^2-bench-Verified domains. The Direct + Delegated configuration achieves higher mean scores than Terminal GRPO and adapted ToolRL, with Pass^5 gains of 8.2 and 7.0 percentage points over Terminal GRPO on Retail and Telecom, respectively. On Telecom, the only domain with user-side state rewards, factual delegated supervision improves on Direct-only, and Online Contextual DCE further improves repeated-trial success at comparable single-trial performance. We further assess DCE by estimating credit adjustments from one subset of neutral replays and validating them on another. User-side progress assigned lower credit is also more likely to recur under the neutral message in the validation replays.

Our contributions are:

- We formulate role-decomposed state-grounded process rewards that distinguish assistant-side progress from factual user-side progress toward task-specific targets, and integrate both into assistant policy optimization.
- We introduce delegated credit estimation (DCE) to refine factual delegated credit through neutral-message replay. Split-replay diagnostics show that contextual DCE identifies user-side progress that recurs under the neutral message and selectively reduces its credit.
- Experiments across three τ^2-bench-Verified domains support the effectiveness of the framework, with additional gains in repeated-trial success from Online Contextual DCE on Telecom. Ablations support selective auxiliary-credit routing and show benefits from process supervision beyond risk penalties.

2 Related Work

2.1 Interactive Tool Use and Dual-Control Environments

Tool-use methods interleave reasoning and action (Yao et al., 2023) or learn API calls through self-supervision (Schick et al., 2023). API-Bank and ToolSandbox evaluate tool-augmented dialogue and intermediate milestones (Li et al., 2023; Lu et al., 2025), whereas τ-bench emphasizes final-state correctness and consistency across repeated trials (Yao et al., 2024). τ^2-Bench extends this evaluation to dual-control interaction, where some required operations are available only through the user's tools (Barres et al., 2025). Communication therefore becomes part of task execution, alongside the assistant's own tool use. Synthetic instruction-tuning data broaden coverage across APIs and invocation patterns (Qin et al., 2024; Liu et al., 2025). Interactive RL methods additionally learn from task execution. MUA-RL includes a simulated user in multi-turn training rollouts (Zhao et al., 2025). AutoForge expands training coverage through environment synthesis and defines success using target states obtained from reference actions (Cai et al., 2025). These approaches provide interactive experience and terminal supervision; intermediate credit assignment remains a separate question.

2.2 Process Supervision for Tool Agents

Reasoning work derives process supervision from human annotations and learned verifiers (Uesato et al., 2022; Lightman et al., 2023) or automatically generated labels (Wang et al., 2024). For tool agents, ToolRL combines format rewards with reference-based matching of tool names and arguments (Qian et al., 2025), while AgentPRM learns value estimates from Monte Carlo returns (Choudhury, 2025). Verifiable intermediate rewards also support turn-level advantage estimation (Zeng et al., 2025). SENTINEL combines task-success rewards with checks and penalties for invalid or degenerate tool use (Wang et al., 2026). TRCA combines executable transition checks with cumulative coverage rewards for positive conditions when first satisfied (Zhang et al., 2026). GiGPO combines episode-level advantages with local comparisons among actions taken from recurring environment states (Feng et al., 2025). Our rewards track target attainment and subsequent

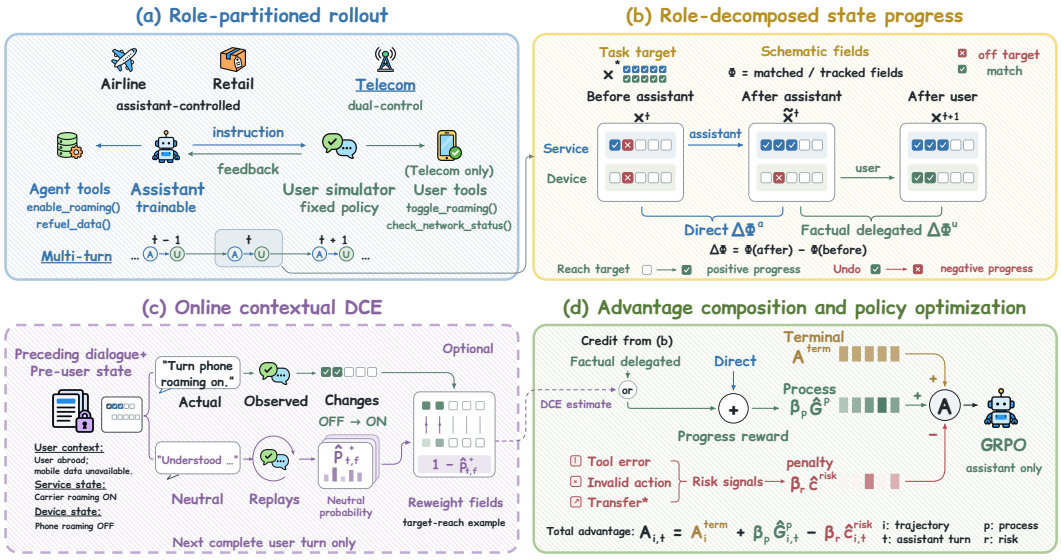

Figure 1: Overview of our framework. (a) Role-partitioned rollouts distinguish assistant-controlled state changes from the user-side device actions exposed in Telecom. (b) Target state potentials yield direct and factual delegated progress. (c) Optional Online Contextual DCE compares the observed factual transition with neutral replays from the same prefix and pre-user state, replacing only the delegated channel. (d) Terminal, process, and risk terms form a turn-level combined advantage. Terminal credit applies to all trainable assistant tokens, while process and risk credit is restricted to tool-call and communication tokens. Panels (a), (b), and (d) form the default training path.

reversals separately for assistant- and user-controlled fields, yielding direct and factual delegated progress.

2.3 User Feedback and Counterfactual Credit Assignment

User feedback can also provide local learning signals. FACA assigns credit using hidden reaction labels generated by the user simulator for the subsequent turn (Zhao et al., 2026). InfoPO measures the change in the likelihood of the assistant's next action when user feedback is retained versus masked (Kong et al., 2026). Our factual delegated signal instead measures task-relevant state changes. Counterfactual methods use future-conditioned baselines to separate action effects from exogenous variation (Mesnard et al., 2021). In cooperative learning, counterfactual baselines and difference rewards address agent-specific credit (Foerster et al., 2018; Castellini et al., 2020), while counterfactual effect decomposition analyzes how actions influence later outcomes through agents and state transitions (Triantafyllou et al., 2025). DCE adjusts credit for observed user-side progress by replacing the current assistant message with a neutral acknowledgement and replaying the next user turn.

3 Role-Decomposed State-Grounded Process Rewards

This section develops state-grounded process rewards for dual-control tool interactions. We first formulate the interaction and define a target-state potential, then derive direct and factual delegated progress together with an operational cost. We subsequently introduce delegated credit estimation (DCE) as an optional adjustment to user-side progress.

3.1 Dual-Control Problem Formulation

We consider interactions between an assistant policy π_θ and a user simulator π_U. In the Telecom domain of τ^2-bench, the mutable state is partitioned as $x = (x^a, x^u)$ into assistant-controlled service state and user-controlled device state as shown in Figure 1(a). Each task q specifies an initial state

x_0. Airline and Retail have no mutable user-side component. Only the assistant is optimized; the user simulator has fixed policy but generates responses and tool calls conditioned on the task context, dialogue, and available user-side observations.

At assistant turn t, y_t denotes assistant response containing text or tool calls, and v_t denotes the complete user turn triggered by that response. Starting from state x_t, the interaction proceeds through:

$$\widetilde{x}_t = T_a(x_t, y_t), \qquad x_{t+1} = T_u(\widetilde{x}_t, v_t). \tag{1}$$

where $\widetilde{x}_t$ is the state after executing the assistant's tool calls and x_{t+1} is the state after the ensuing user interaction. When no user turn occurs, $v_t = \varnothing$ and $T_u(\widetilde{x}_t, \varnothing) = \widetilde{x}_t$. Within this partition, assistant operations change only x^a, and user operations change only x^u.

3.2 Target-State Potential

For each task, an independent environment is restored to the initial state x_0. The reference actions are then executed in order, using their specified roles. The resulting state $x^* = (x^{a,*}, x^{u,*})$ defines the target values for supervision. The policy may reach these values through a different sequence of actions.

We flatten each state component into scalar fields and retain those whose initial and target values differ:

$$\mathcal{R}_s = \{f : x_0^s[f] \neq x^{s,*}[f]\}, \qquad s \in \{a, u\}. \tag{2}$$

For each role, the potential measures the fraction of retained fields that currently match their target values:

$$\Phi_s(x_t) = \begin{cases} \dfrac{1}{|\mathcal{R}_s|} \displaystyle\sum_{f \in \mathcal{R}_s} \mathbf{1}[x_t^s[f] = x^{s,*}[f]], & |\mathcal{R}_s| > 0, \\ 0, & |\mathcal{R}_s| = 0. \end{cases} \tag{3}$$

For a retained field, reaching its target value increases the potential, leaving it decreases the potential, and changes between non-target values leave it unchanged. Since the potential tracks only initial target differences, it does not cover target-invariant fields or communication requirements. It provides partial supervision alongside the terminal reward. State restoration, field handling, and further limitations are detailed in Appendix A.

3.3 Turn-Level Progress and Risk Signals

Direct and Delegated Progress. We use undiscounted potential differences (Ng et al., 1999) to measure progress over the role-specific transitions in Equation 1 (Figure 1(b)):

$$\begin{aligned} \Delta\Phi_t^a &= \Phi_a(\widetilde{x}_t) - \Phi_a(x_t) = \Phi_a(x_{t+1}) - \Phi_a(x_t), \\ \Delta\Phi_t^u &= \Phi_u(x_{t+1}) - \Phi_u(\widetilde{x}_t) = \Phi_u(x_{t+1}) - \Phi_u(x_t). \end{aligned} \tag{4}$$

Note that the second equality in both lines holds because assistant actions do not modify x^u and user actions do not modify x^a. These changes define the raw progress components and their weighted process reward:

$$\begin{aligned} g_t^{\mathrm{dir}} = \Delta\Phi_t^a, \qquad g_t^{\mathrm{del,fact}} = \Delta\Phi_t^u, \\ r_t^p = \alpha_{\mathrm{dir}} g_t^{\mathrm{dir}} + \alpha_{\mathrm{del}} g_t^{\mathrm{del,fact}}, \end{aligned} \tag{5}$$

where the nonnegative weights α_{dir} and α_{del} control the relative contributions of direct and delegated progress. Factual delegated progress is assigned to the assistant response that precedes the user turn. For each role, the undiscounted sum of raw factual signals equals the net change in the corresponding role's potential, so a subsequent reversal offsets earlier progress (Appendix A).

Risk Signals. Tool-call errors and invalid actions can leave all tracked fields unchanged and thus receive zero progress credit. We penalize these events through a separate nonnegative risk cost. At turn t, let e_t count new assistant tool-call errors, ι_t indicate an invalid or unparsable action, and n_t^{tr} count new transfer calls when transfer is absent from the task's reference actions. We define:

$$c_t^{\mathrm{risk}} = \lambda_{\mathrm{err}} e_t + \lambda_{\mathrm{inv}} \iota_t + \lambda_{\mathrm{tr}} n_t^{\mathrm{tr}}, \tag{6}$$

where $\lambda_{\mathrm{err}}, \lambda_{\mathrm{inv}}, \lambda_{\mathrm{tr}} \geq 0$ weight tool-call errors, invalid actions, and transfer calls, respectively. All applicable costs are added within each assistant turn. Event-detection rules are given in Appendix B, and coefficient settings in Appendix E.3.

3.4 DELEGATED CREDIT ESTIMATION

Factual delegated progress is the default signal in Equation 5, but need not depend on the current instruction. DCE adjusts this signal through user-turn replay, replacing $g_t^{\mathrm{del,fact}}$ with $g_t^{\mathrm{del,on}}$ or $g_t^{\mathrm{del,off}}$. For each relevant field f, the observed factual transition is:

$$z_{t,f}^A = \mathbf{1}\{x_{t+1}^u[f] = x^{u,*}[f]\} - \mathbf{1}\{\widetilde{x}_t^u[f] = x^{u,*}[f]\},$$
$$g_t^{\mathrm{del,fact}} = \frac{1}{|\mathcal{R}_u|} \sum_{f \in \mathcal{R}_u} z_{t,f}^A, \tag{7}$$

where $z_{t,f}^A \in \{-1, 0, +1\}$ records regression, unchanged target status, or target attainment. All delegated signals are zero when $\mathcal{R}_u = \varnothing$. Neutral replay replaces the selected assistant message in full with a fixed acknowledgement, preserving the preceding dialogue and pre-user state $\widetilde{x}_t$. Each replay covers the ensuing user turn, including user-side tool calls. The resulting frequencies describe state changes under this message replacement, context, and simulator.

Online Contextual DCE. Online Contextual DCE (Figure 1(c)) triggers replay when a user-side target field changes its target-match status. It retains the observed factual branch A and independently samples n neutral-branch B replays with fresh simulator randomness. The adjusted signal is:

$$\widetilde{g}_{t,f} = \begin{cases} 1 - \widehat{p}_{t,f}^{+}, & z_{t,f}^A = +1, \\ 0, & z_{t,f}^A = 0, \\ -\big(1 - \widehat{p}_{t,f}^{-}\big), & z_{t,f}^A = -1, \end{cases}$$
$$g_t^{\mathrm{del,on}} = \frac{1}{|\mathcal{R}_u|} \sum_{f \in \mathcal{R}_u} \widetilde{g}_{t,f}, \tag{8}$$

where $\widehat{p}_{t,f}^{+}$ and $\widehat{p}_{t,f}^{-}$ are the fractions of neutral replays in which field f moves onto and off its target, respectively. Conditional on an observed gain, $1 - \widehat{p}_{t,f}^{+}$ estimates its probability of non-recurrence under neutral replay; the regression weight has the analogous interpretation. The rule preserves the factual sign and assigns zero to unchanged fields, adjusting observed credit according to replay recurrence. Appendix D gives the trigger, estimator properties, and diagnostics.

Offline Field-DCE. Before policy training, Offline Field-DCE independently replays both the original-message branch A and neutral branch B from the same dialogue and pre-user state. For field f, the calibration set $\mathcal{I}_f$ contains recorded turns whose current instruction targets that field; both branches use this set. The field weights and adjusted signal are:

$$w_f^{\mathrm{off}} = \mathrm{clip}_{[0,1]}\left(1 - \frac{\widehat{P}_B(f)}{\widehat{P}_A(f)}\right),$$
$$g_t^{\mathrm{del,off}} = \frac{1}{|\mathcal{R}_u|} \sum_{f \in \mathcal{R}_u} w_f^{\mathrm{off}} z_{t,f}^A, \tag{9}$$

where $\widehat{P}_A(f)$ and $\widehat{P}_B(f)$ are the fractions of replays over $\mathcal{I}_f$ that end with f at its target under A and B, respectively. We set $w_f^{\mathrm{off}} = 0$ when $\mathcal{I}_f = \varnothing$ or $\widehat{P}_A(f) = 0$. Each frozen weight scales both gains and regressions during training, without further DCE replay. Appendix C details replay selection, probability estimation, and the calibrated field weights.

4 FROM TURN-LEVEL REWARDS TO GRPO OPTIMIZATION

Figure 1(d) summarizes how the turn-level signals enter policy optimization. Process returns and operational costs augment the terminal GRPO advantage, with separate temporal treatment and normalization before assignment to assistant tokens.

4.1 Process Returns and Token Routing

For trajectory i, let $r^p_{i,t}$ be the process reward in Equation 5, including the DCE substitution when enabled. Subsequent progress contributes to earlier assistant decisions through the sampled discounted return (Sutton et al., 1999):

$$G^p_{i,t} = \sum_{k \geq t} \gamma^{k-t} r^p_{i,k}, \tag{10}$$

where γ discounts across assistant turns. Risk costs remain on the turn in which the penalized event occurs.

State transitions assess executed actions and user-facing instructions, but do not identify the contribution of individual reasoning steps. By default, we assign process and risk signals only to assistant tool-call and communication tokens. Reasoning tokens retain only terminal supervision; user, environment, and tool-response tokens are excluded from optimization.

4.2 Normalization and Advantage Composition

A sibling group contains trajectories sampled for the same task. Let A^{term}_i be the GRPO advantage computed from their binary terminal rewards (Shao et al., 2024). Baselines are commonly used for variance reduction in policy-gradient estimation (Greensmith et al., 2004). Here, each complete group g uses a shared empirical baseline b_g, the token-weighted mean of process returns over credited tokens across all turns and trajectories. We set $\bar{G}^p_{i,t} = G^p_{i,t} - b_g$ for groups containing failures and $\bar{G}^p_{i,t} = 0$ for all-success groups, so process returns do not rank trajectories that all satisfy the terminal criterion. Risk penalties remain active. For groups containing failures, subtracting the shared baseline preserves within-group differences in discounted returns before normalization and clipping. We normalize and clip the process return as:

$$\widehat{G}^p_{i,t} = \text{clip}\left(\frac{\bar{G}^p_{i,t}}{\sqrt{\text{RMS}(\bar{G}^p)^2 + \varepsilon_{\text{rms}}}}, -c, c \right), \tag{11}$$

where RMS is computed over credited-token entries from complete groups in the batch, including those zeroed by the all-success gate. Tokens outside the auxiliary routing mask are excluded. Here $c > 0$ bounds the normalized signal and $\varepsilon_{\text{rms}} > 0$ stabilizes the denominator.

Risk costs use a separate RMS over all turns in the batch, including zero-cost turns:

$$\widehat{c}^{\text{risk}}_{i,t} = \text{clip}\left(\frac{c^{\text{risk}}_{i,t}}{\sqrt{\text{RMS}(c^{\text{risk}})^2 + \varepsilon_{\text{rms}}}}, 0, c \right). \tag{12}$$

Without centering, zero-cost turns retain zero penalty, and lower-than-average costs cannot become auxiliary bonuses. RMS normalization controls the overall scale of each auxiliary signal, while clipping limits the contribution of large normalized returns or costs to the token advantage.

The turn-level combined advantage is:

$$A_{i,t} = A^{\text{term}}_i + \beta_p \widehat{G}^p_{i,t} - \beta_r \widehat{c}^{\text{risk}}_{i,t}, \tag{13}$$

where the nonnegative coefficients β_p and β_r control the contributions of process feedback and risk penalties, respectively. For each trainable assistant token ℓ, let $t(\ell)$ denote the turn containing that token. By default, we assign $A_{i,\ell} = A_{i,t(\ell)}$ to tool-call and communication tokens, and $A_{i,\ell} = A^{\text{term}}_i$ to reasoning tokens. In all-failure groups, terminal advantages vanish, while differences in process returns can still distinguish trajectories.

4.3 Policy Optimization

For generated token $o_{i,\ell}$ with prefix $c_{i,\ell}$, define the ratio to the rollout policy $\pi_{\theta_{\text{old}}}$:

$$\rho_{i,\ell}(\theta) = \frac{\pi_\theta(o_{i,\ell} \mid c_{i,\ell})}{\pi_{\theta_{\text{old}}}(o_{i,\ell} \mid c_{i,\ell})}. \tag{14}$$

Method	Airline			Retail			Telecom		
	Pass^1	Pass^3	Pass^5	Pass^1	Pass^3	Pass^5	Pass^1	Pass^3	Pass^5
Qwen3.5-9B Base	74.0	58.4	51.3	76.5	55.9	45.0	90.4	76.7	67.8
Terminal GRPO	75.9	63.7	57.3	75.9	54.9	43.0	94.0	83.8	75.7
ToolRL (adapted)	72.5	59.3	53.3	76.9	56.3	45.9	94.5	85.1	77.5
Direct-only	75.2	62.8	56.7	76.7	57.1	47.7	95.4	87.4	80.7
Direct + Delegated	**76.3**	**63.8**	**58.0**	**80.3**	**61.6**	**51.2**	**96.2**	**89.2**	**82.7**

Table 1: Policy performance on the three τ^2-bench-Verified domains (%). Scores are averaged over three evaluation runs with five trials per task. Bold marks the best value for each metric within each domain, including ties.

We use $\bar{\rho}_{i,\ell}(\theta) = \mathrm{clip}(\rho_{i,\ell}(\theta), 1-\epsilon_{\mathrm{low}}, 1+\epsilon_{\mathrm{high}})$ to allow asymmetric clipping (Yu et al., 2025). Over the set $\mathcal{T}$ of trainable assistant tokens, we use the clipped surrogate (Schulman et al., 2017):

$$\mathcal{L}_{\mathrm{policy}}(\theta) = -\frac{1}{|\mathcal{T}|} \sum_{(i,\ell)\in\mathcal{T}} \min(\rho_{i,\ell}(\theta)A_{i,\ell}, \bar{\rho}_{i,\ell}(\theta)A_{i,\ell}) + \beta_{\mathrm{KL}}\, \mathcal{K}(\pi_\theta, \pi_{\mathrm{ref}}), \tag{15}$$

where $\mathcal{K}$ is the KL regularizer relative to the reference policy π_{ref} (Ziegler et al., 2019). The token advantages $A_{i,\ell}$ follow the assignment rule after Equation 13. Appendix B details the masks and aggregation, and Appendix E.3 reports the optimization settings.

5 Experiments

We evaluate whether state-grounded process rewards improve multi-turn tool agents, focusing on delegated supervision and DCE's effects on policy learning, followed by ablations of auxiliary-credit routing and process strength.

5.1 Experimental Setup

Benchmark and models. We evaluate on the Airline, Retail, and Telecom domains of the corrected, human-verified τ^2-bench-Verified release (Barres et al., 2025; Cuadron et al., 2025). All RL variants initialize from the same post-trained Qwen3.5-9B (Qwen Team, 2026b) checkpoint and train jointly on the same three-domain data for 120 steps before domain-wise evaluation. All methods share a fixed Qwen3.6-35B-A3B (Qwen Team, 2026a) user-simulator policy and evaluation protocol. Appendix E details data sources, splits, and training settings.

Compared methods. **Qwen3.5-9B Base** is the initial checkpoint without additional RL. **Terminal GRPO** uses binary terminal success rewards. **ToolRL (adapted)** (Qian et al., 2025) adds process rewards matching assistant and user tool calls against reference actions by name and arguments. **Direct-only** adds assistant-side state progress and operational-risk penalties to terminal supervision; **Direct + Delegated** further adds factual user-side progress with the same risk setting. Appendix E.2 specifies rewards, routing, and gating mechanism.

Evaluation protocol. We use the benchmark's terminal evaluator and conduct five trials per task in each evaluation run. Pass^1 measures average single-trial success; Pass^k estimates the probability that k trials sampled without replacement all succeed (Yao et al., 2024). Each checkpoint undergoes three evaluation repeats with the same master seed shared across runs and methods. We report their means, not averages over independent training runs. Appendices E.4 and E.5 provide the protocol and variability statistics.

5.2 Policy Performance across Domains

Table 1 shows that Direct + Delegated achieves the highest mean scores across all domains and metrics among the five compared methods. Relative to Terminal GRPO, Pass^5 improves by 8.2, 7.0, and 0.7 percentage points on Retail, Telecom, and Airline, respectively. Retail and Telecom show larger gains in Pass^5 than Pass^1, indicating improved repeated-trial success. Pass^5 also exceeds

Delegated signal	Pass[1]	Pass[2]	Pass[3]	Pass[4]	Pass[5]
Factual progress	**96.2**	**92.6**	89.2	85.9	82.7
Offline Field-DCE	95.7	92.0	88.6	85.8	83.3
Online Contextual DCE	96.0	92.5	**89.5**	**87.0**	**84.8**

Table 2: Effect of DCE on Telecom policy performance (%). Scores are averaged over three evaluation runs. Factual progress denotes the Direct + Delegated policy reported in Table 1. Bold marks the best value for each metric.

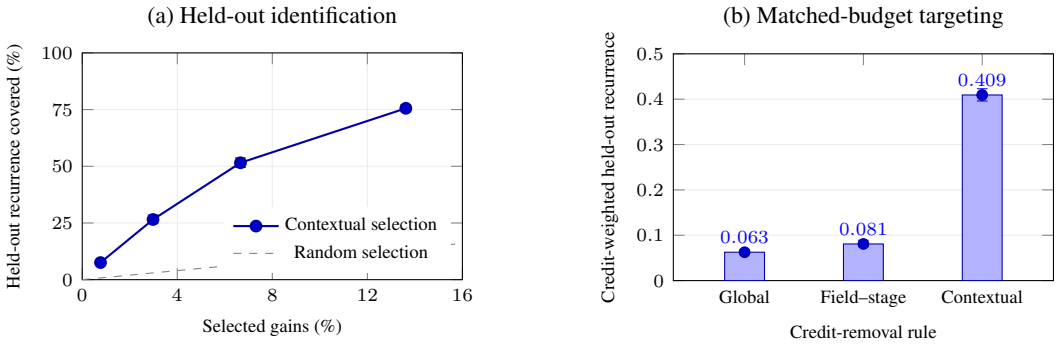

Figure 2: Contextual DCE identifies and targets neutrally reproducible progress. Four of each user turn's eight recorded replays determine field-level selection and credit removal; four held-out replays measure recurrence. (a) Gains selected at four recurrence-frequency thresholds, connected by lines. (b) Credit-weighted recurrence under matched removal budgets. Error bars: 95% task-cluster bootstrap intervals (Appendix D).

adapted ToolRL by 4.7–5.3 points across domains. These are complete-configuration comparisons: reward construction, routing, gating, and risk penalties differ.

The comparison with Direct-only examines the incremental value of factual delegated progress on Telecom, where user actions can modify task-relevant state. Direct-only already raises Pass[5] from 75.7 to 80.7 relative to Terminal GRPO; adding delegated progress raises it further to 82.7, alongside a 0.8-point increase in Pass[1]. The direct-progress and risk settings are unchanged between these two variants. These gains support crediting user-side state changes alongside assistant-side progress during joint training, providing supervision for task advances that assistant-only state tracking would omit. Since only Telecom supplies nonzero delegated rewards, the gains on Airline and Retail reflect changes in the shared, jointly trained policy.

5.3 Effect of DCE

Table 2 compares factual progress and both DCE variants on Telecom, using the same direct-progress and risk settings. Online Contextual DCE increases Pass[5] from 82.7 to 84.8 at comparable single-trial performance (96.0 versus 96.2 Pass[1]). Offline Field-DCE reaches 83.3 Pass[5]; the two DCE variants use different replay sources and settings (Appendix E.3).

Identifying neutrally reproducible progress. We analyze 59,294 transitions of user-side fields from off-target to target values across 1,283 training tasks. Each user turn's eight recorded neutral replays are split into four for estimating credit adjustments and four for validation. Gains reproduced at least once in the first set have a validation recurrence rate of 33.7%, compared with 1.7% for the remaining gains. This separation shows that recurrence estimates predict outcomes in independent replays; Figure 2(a) summarizes identification across selection thresholds.

Targeting credit beyond uniform attenuation. Using the first four replays, we compare contextual DCE with a global constant discount and constants specific to each field and training stage. All rules remove the same total credit; the field–stage control also matches removal within each field–stage combination. We measure validation recurrence averaged with weights proportional to removed credit. Contextual DCE scores 0.409, versus 0.063 and 0.081 for the global and field–stage

		Airline			Retail			Telecom		
Routing	β_p	Pass^1	Pass^3	Pass^5	Pass^1	Pass^3	Pass^5	Pass^1	Pass^3	Pass^5
Routed	0.2	**76.3**	**63.8**	**58.0**	**80.3**	**61.6**	**51.2**	**96.2**	**89.2**	**82.7**
All tokens	0.2	73.3	60.0	53.3	77.8	58.2	45.9	94.6	86.4	80.1
Routed	0	74.0	62.3	56.0	75.2	54.9	45.0	94.8	85.7	78.0
Routed	0.5	75.9	64.2	58.0	78.7	59.3	49.7	96.1	89.0	82.8

Table 3: Routing and process-strength ablations across domains (%; means over three evaluation runs). The first row repeats Direct + Delegated from Table 1. Routed restricts process and risk credit to tool-call and communication tokens; All tokens applies both to all trainable assistant tokens, including reasoning. Bold marks the better value between these two routing settings. The Routed rows vary β_p with $\beta_r = 0.1$ fixed.

controls (Figure 2(b)). Thus, contextual replay concentrates credit removal on neutrally reproducible progress beyond field and stage averages. These controls assess credit allocation on fixed trajectories, not separately trained policies. Appendix D provides full diagnostics and replay audits.

5.4 Ablation Studies

We examine how auxiliary-credit routing and process-reward strength affect performance across Airline, Retail, and Telecom (Table 3).

Auxiliary-credit routing. We apply both process and risk credit to tool-call and communication tokens (Routed) or all trainable assistant tokens, including reasoning (All tokens), with coefficients and terminal supervision fixed. Routed improves all reported mean metrics across domains. Pass^5 gains of 4.7, 5.3, and 2.6 percentage points on Airline, Retail, and Telecom exceed the respective Pass^1 gains, supporting selective auxiliary-credit routing.

Process-reward strength. We vary $\beta_p \in \{0, 0.2, 0.5\}$ with β_r, routing, and other settings fixed; $\beta_p = 0$ yields Terminal + Risk. Both nonzero weights improve Pass^1, Pass^3, and Pass^5 over this baseline. At the primary weight $\beta_p = 0.2$, Pass^5 gains are 2.0, 6.2, and 4.7 percentage points on Airline, Retail, and Telecom. Raising β_p to 0.5 leaves Airline and Telecom scores similar but lowers Retail Pass^5 from 51.2 to 49.7, still above 45.0 without process supervision. Process feedback therefore adds value beyond risk penalties, while larger weights need not improve performance. At $\gamma = 1$, the raw factual process return combines terminal partial completion with a current-state term (Appendix A); this ablation establishes the value of the combined signal without isolating their contributions.

6 Conclusion and Limitations

We studied process supervision for tool-use tasks where both assistant and user actions contribute to task completion. Our framework uses role-decomposed state rewards to measure progress, alongside contextual DCE to adjust delegated credit via neutral replay. This approach improves mean performance over Terminal GRPO and ToolRL across three τ^2-bench-Verified domains, notably enhancing repeated-trial success in Telecom. Diagnostics confirm that DCE effectively identifies and discounts neutrally reproducible progress. Together, these findings support learning from factual user-side progress and refining its credit through contextual replay.

Our approach requires structured states and task-specific reference actions. The evaluation of delegated credit is limited to Telecom with a single user simulator, leaving generalization to other simulators and human users untested. Online Contextual DCE also requires repeated user-turn simulations, which add computational overhead and reduce training throughput. Future work will evaluate robustness across user simulators and investigate adaptive replay budgets and learned credit estimators to reduce this overhead.

AI USE STATEMENT

The algorithmic ideas, experimental design, and synthetic-data generation methodology were developed by the authors. We used generative AI tools to write the implementation code for the proposed methods, experimental pipelines, and data-generation procedure. We also used generative models to produce additional synthetic training data for τ^2-RL according to the author-designed procedure, supplementing the limited available training data. These tools further assisted with the analysis and interpretation of experimental results. All sections of the manuscript and the figures were developed through iterative AI-assisted generation. The authors verified the generated code, synthetic data, and AI-assisted analyses, and thoroughly reviewed and revised the manuscript and figures. The authors take full responsibility for the final content of this work, including its methods, code, synthetic data, experimental results, text, and figures.

REPRODUCIBILITY STATEMENT

Sections 3 and 4 describe the state-grounded rewards, delegated credit estimation, and policy optimization procedure. Appendix B provides implementation details of advantage construction and token-level credit routing. Appendices C and D report the offline field weights and recorded-replay diagnostics, including the split-replay design and matched-budget controls. Appendix E.2 specifies the compared systems and the ToolRL adaptation, while Appendix E.4 documents the evaluation protocol, random-seed mechanism, and aggregation of repeated evaluations. To facilitate reproduction, we will provide a public repository containing implementations of the proposed methods and compared baselines, training and evaluation scripts and configurations, synthetic training data and its generation code, and task splits upon acceptance.

REFERENCES

Ibrahim Abdelaziz, Asim Munawar, Kinjal Basu, Maxwell Crouse, Chulaka Gunasekara, Suneet Katrekar, and Pavan Kapanipathi. Synthesize and reward: Reinforcement learning for multi-step tool use in live environments. *arXiv preprint arXiv:2606.03892*, 2026. URL https://arxiv.org/abs/2606.03892.

Victor Barres, Honghua Dong, Soham Ray, Xujie Si, and Karthik Narasimhan. τ^2-Bench: Evaluating conversational agents in a dual-control environment. *arXiv preprint arXiv:2506.07982*, 2025.

Shihao Cai, Runnan Fang, Jialong Wu, Baixuan Li, Xinyu Wang, Yong Jiang, Liangcai Su, Liwen Zhang, Wenbiao Yin, Zhen Zhang, Fuli Feng, Pengjun Xie, and Xiaobin Wang. AutoForge: Automated environment synthesis for agentic reinforcement learning. *arXiv preprint arXiv:2512.22857*, 2025. URL https://arxiv.org/abs/2512.22857.

Jacopo Castellini, Sam Devlin, Frans A. Oliehoek, and Rahul Savani. Difference rewards policy gradients. *arXiv preprint arXiv:2012.11258*, 2020.

Sanjiban Choudhury. Process reward models for LLM agents: Practical framework and directions. *arXiv preprint arXiv:2502.10325*, 2025.

Alejandro Cuadron, Pengfei Yu, Yang Liu, and Arpit Gupta. Saber: Small actions, big errors - safeguarding mutating steps in llm agents, 2025. URL https://arxiv.org/abs/2512.07850.

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*, volume 38, 2025. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/420c9f777c0b4f78d515e53cf74d58b2-Abstract-Conference.html.

C. A. Field and A. H. Welsh. Bootstrapping clustered data. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 69(3):369–390, 2007. doi: 10.1111/j.1467-9868.2007.00593.x. URL https://doi.org/10.1111/j.1467-9868.2007.00593.x.

Jakob Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. Counterfactual multi-agent policy gradients. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2018.

Jiaxuan Gao, Jiaao Chen, Chuyi He, Shusheng Xu, Di Jin, and Yi Wu. From self-evolving synthetic data to verifiable-reward RL: Post-training multi-turn interactive tool-using agents. *arXiv preprint arXiv:2601.22607*, 2026. URL https://arxiv.org/abs/2601.22607.

Evan Greensmith, Peter L. Bartlett, and Jonathan Baxter. Variance reduction techniques for gradient estimates in reinforcement learning. *Journal of Machine Learning Research*, 5:1471–1530, 2004. URL https://www.jmlr.org/papers/v5/greensmith04a.html.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015. URL https://arxiv.org/abs/1412.6980.

Fanqi Kong, Jiayi Zhang, Mingyi Deng, Chenglin Wu, Yuyu Luo, and Bang Liu. InfoPO: Information-driven policy optimization for user-centric agents. *arXiv preprint arXiv:2603.00656*, 2026. URL https://arxiv.org/abs/2603.00656.

Minghao Li, Yingxiu Zhao, Bowen Yu, Feifan Song, Hangyu Li, Haiyang Yu, Zhoujun Li, Fei Huang, and Yongbin Li. API-Bank: A comprehensive benchmark for tool-augmented LLMs. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 3102–3116. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.emnlp-main.187. URL https://aclanthology.org/2023.emnlp-main.187/.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. *arXiv preprint arXiv:2305.20050*, 2023.

Weiwen Liu, Xu Huang, Xingshan Zeng, Xinlong Hao, Shuai Yu, Dexun Li, Shuai Wang, Weinan Gan, Zhengying Liu, Yuanqing Yu, Zezhong Wang, Yuxian Wang, Wu Ning, Yutai Hou, Bin Wang, Chuhan Wu, Xinzhi Wang, Yong Liu, Yasheng Wang, Duyu Tang, Dandan Tu, Lifeng Shang, Xin Jiang, Ruiming Tang, Defu Lian, Qun Liu, and Enhong Chen. ToolACE: Winning the points of LLM function calling. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/663865ea167425c6c562cb0b6bcf76c7-Abstract-Conference.html.

Jiarui Lu, Thomas Holleis, Yizhe Zhang, Bernhard Aumayer, Feng Nan, Haoping Bai, Shuang Ma, Shen Ma, Mengyu Li, Guoli Yin, Zirui Wang, and Ruoming Pang. ToolSandbox: A stateful, conversational, interactive evaluation benchmark for LLM tool use capabilities. In Luis Chiruzzo, Alan Ritter, and Lu Wang (eds.), *Findings of the Association for Computational Linguistics: NAACL 2025*, pp. 1160–1183, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.findings-naacl.65. URL https://aclanthology.org/2025.findings-naacl.65/.

Thomas Mesnard, Theophane Weber, Fabio Viola, Shantanu Thakoor, Alaa Saade, Anna Harutyunyan, Will Dabney, Thomas S. Stepleton, Nicolas Heess, Arthur Guez, Eric Moulines, Marcus Hutter, Lars Buesing, and Remi Munos. Counterfactual credit assignment in model-free reinforcement learning. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 7654–7664. PMLR, 2021. URL https://proceedings.mlr.press/v139/mesnard21a.html.

Andrew Y. Ng, Daishi Harada, and Stuart Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *Proceedings of the Sixteenth International Conference on Machine Learning*, 1999. URL https://people.eecs.berkeley.edu/~russell/papers/icml99-shaping.pdf.

Cheng Qian, Emre Can Acikgoz, Qi He, Hongru Wang, Xiusi Chen, Dilek Hakkani-Tür, Gokhan Tur, and Heng Ji. ToolRL: Reward is all tool learning needs. In *Advances in Neural Information Processing Systems*, volume 38, 2025. URL

https://proceedings.neurips.cc/paper_files/paper/2025/hash/97c5b2707228e7e3fb67e4ecc2e0e607-Abstract-Conference.html.

Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, Sihan Zhao, Lauren Hong, Runchu Tian, Ruobing Xie, Jie Zhou, Mark Gerstein, Dahai Li, Zhiyuan Liu, and Maosong Sun. ToolLLM: Facilitating large language models to master 16000+ real-world APIs. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/28e50ee5b72e90b50e7196fde8ea260e-Abstract-Conference.html.

Qwen Team. Qwen3.6-35B-A3B: Agentic coding power, now open to all, April 2026a. URL https://qwen.ai/blog?id=qwen3.6-35b-a3b.

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

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://proceedings.neurips.cc/paper/2023/hash/d842425e4bf79ba039352da0f658a906-Abstract-Conference.html.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017. URL https://arxiv.org/abs/1707.06347.

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.

Richard S. Sutton, David A. McAllester, Satinder P. Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems*, volume 12, pp. 1057–1063, 1999. URL https://proceedings.neurips.cc/paper/1999/hash/464d828b85b0bed98e80ade0a5c43b0f-Abstract.html.

Jin Tian and Judea Pearl. Probabilities of causation: Bounds and identification. *Annals of Mathematics and Artificial Intelligence*, 28:287–313, 2000. URL https://ftp.cs.ucla.edu/pub/stat_ser/r271-A.pdf.

Stelios Triantafyllou, Aleksa Sukovic, Yasaman Zolfimoselo, and Goran Radanovic. Counterfactual effect decomposition in multi-agent sequential decision making. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 60072–60098. PMLR, 2025. URL https://proceedings.mlr.press/v267/triantafyllou25a.html.

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

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. Math-Shepherd: Verify and reinforce LLMs step-by-step without human annotations. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9426–9439, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.510. URL https://aclanthology.org/2024.acl-long.510/.

Ziyi Wang, Yuxuan Lu, Yimeng Zhang, Qun Liu, Chen Luo, Jiri Gesi, Hanqing Lu, Yisi Sang, Manling Li, Jing Huang, and Dakuo Wang. SENTINEL: Failure-driven reinforcement learning for training tool-using language model agents. *arXiv preprint arXiv:2606.12908*, 2026. URL https://arxiv.org/abs/2606.12908.

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.

Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik Narasimhan. τ-bench: A benchmark for tool-agent-user interaction in real-world domains. *arXiv preprint arXiv:2406.12045*, 2024.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, Haibin Lin, Zhiqi Lin, Bole Ma, Guangming Sheng, Yuxuan Tong, Chi Zhang, Mofan Zhang, Wang Zhang, Hang Zhu, Jinhua Zhu, Jiaze Chen, Jiangjie Chen, Chengyi Wang, Hongli Yu, Yuxuan Song, Xiangpeng Wei, Hao Zhou, Jingjing Liu, Wei-Ying Ma, Ya-Qin Zhang, Lin Yan, Mu Qiao, Yonghui Wu, and Mingxuan Wang. DAPO: An open-source LLM reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025. URL https://arxiv.org/abs/2503.14476.

Siliang Zeng, Quan Wei, William Brown, Oana Frunza, Yuriy Nevmyvaka, and Mingyi Hong. Reinforcing multi-turn reasoning in LLM agents via turn-level credit assignment. *arXiv preprint arXiv:2505.11821v1*, 2025. URL https://arxiv.org/abs/2505.11821v1.

Huan Zhang, Mingju Chen, Dongxu Zhou, Can Lv, Heng Chang, Sen Cui, Faguo Wu, and Shiji Zhou. TRCA: Transition-wise rubric credit assignment for long-horizon LLM agents. *arXiv preprint arXiv:2608.16156*, 2026.

Weikang Zhao, Xili Wang, Chengdi Ma, Lingbin Kong, Zhaohua Yang, Mingxiang Tuo, Xiaowei Shi, Yitao Zhai, and Xunliang Cai. MUA-RL: Multi-turn user-interacting agent reinforcement learning for agentic tool use. *arXiv preprint arXiv:2508.18669*, 2025. URL https://arxiv.org/abs/2508.18669.

Yiwen Zhao, Zhihao Wen, Yuchen Mao, Mingxuan Jiang, Yihao Hu, Pan Wang, Xin Zhang, and Wei Wu. Towards better agents for multi-turn user interaction: The next user turn is more than context. *arXiv preprint arXiv:2608.17499*, 2026. URL https://arxiv.org/abs/2608.17499.

Daniel M. Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B. Brown, Alec Radford, Dario Amodei, Paul Christiano, and Geoffrey Irving. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*, 2019. URL https://arxiv.org/abs/1909.08593.

A State-Potential Reward Construction

A.1 State Restoration and Target Construction

For each task, the tracker constructs two fresh environments. The first is restored to the true task start using initialization data, initialization actions, and message history; its assistant and user database dumps define $x_0 = (x_0^a, x_0^u)$. The second is restored identically and then executes the golden actions sequentially with their declared requestors and arguments, producing the canonical target $x^* = (x^{a,*}, x^{u,*})$. This follows the terminal evaluator's gold-state construction, including leaving the replay state unchanged when a call cannot be applied.

The implementation flattens dictionaries into dotted paths and lists or tuples by positional index. It takes the union of initial and target paths and uses a dedicated missing-value sentinel on either side, so insertions and deletions can enter $\mathcal{R}_s$. Scalar values are compared by exact equality. At reset, the tracker evaluates both potentials on the live environment and stores them as the previous snapshot.

The potential in Equation 3 averages binary target-match indicators over the retained fields. For a field in $\mathcal{R}_s$, reaching its target contributes $+1/|\mathcal{R}_s|$ to the potential change, leaving it contributes $-1/|\mathcal{R}_s|$, and moving between non-target values contributes zero. The potential is zero when $\mathcal{R}_s$ is empty.

A.2 Runtime Turn Boundaries and Role Decomposition

A runtime step corresponds to an assistant turn t in Section 3.1: the assistant response and any tool execution, followed, when applicable, by the complete user turn. The tracker compares the resulting state x_{t+1} with the preceding snapshot x_t. Assistant tool calls affect the assistant-controlled

database, whereas user tool calls affect the user-controlled database. Comparing the two turn-boundary snapshots therefore recovers both role-specific potential differences in Equation 4, including the net effect of all tool calls within the user turn.

If potential construction fails, the implementation records the initialization status and emits zero process progress while retaining operational cost.

A.3 Formal Properties and Limitations

Proposition 1 (telescoping). For either state partition $s \in \{a, u\}$, the undiscounted sum of raw factual potential differences over T assistant turns satisfies:

$$\sum_{t=0}^{T-1} \Delta\Phi_t^s = \Phi_s(x_T) - \Phi_s(x_0). \tag{16}$$

Proof. Expanding the differences cancels every intermediate potential. Consequently, a field that enters and later leaves its target contributes zero net factual progress. □

Because $\Phi_s \in [0, 1]$, the change over any contiguous subtrajectory lies in $[-1, 1]$. At the constructed task start, all retained fields differ from their targets, giving $\Phi_s(x_0) = 0$; thus the full-trajectory factual sum lies in $[0, 1]$. With the reported $\gamma = 1$, each suffix sum of these raw differences similarly equals $\Phi_s(x_T) - \Phi_s(x_t)$. This identity concerns raw factual progress and does not imply the same cancellation in the final token advantages after centering, gating, RMS normalization, and clipping.

Relation to terminal partial-state supervision. For fixed role weights, define $F(x) = \alpha_{\mathrm{dir}}\Phi_a(x) + \alpha_{\mathrm{del}}\Phi_u(x)$. At $\gamma = 1$, the factual return is $G^p_{i,t} = F(x_{i,T_i}) - F(x_{i,t})$, whereas a reward supplied only at the terminal state gives the same endpoint score $F(x_{i,T_i})$ to every preceding turn. To compare their centered signals, let C_g subtract the token-weighted group mean in Equation 18, using the same credited-token weights for each argument. Linearity gives:

$$(C_g G^p)_{i,t} = (C_g F_T)_{i,t} - (C_g F_{\mathrm{cur}})_{i,t}, \tag{17}$$

where $(F_T)_{i,t} = F(x_{i,T_i})$ and $(F_{\mathrm{cur}})_{i,t} = F(x_{i,t})$. The all-success gate can multiply both sides without changing this identity. Thus, group centering does not generally remove the current-potential term. Separate RMS normalization and clipping do not preserve this additive decomposition. This identifies how the training signals differ; the reported experiments do not isolate a performance advantage over matched terminal partial-state supervision.

Role decomposition and a joint potential. With the same fixed role weights, $r^p_t = F(x_{t+1}) - F(x_t)$ is exactly the factual role-decomposed reward. Writing it as a single joint potential therefore leaves this reward unchanged. A mean pooled over all fields instead weights the role-specific means in proportion to their field counts, generally changing the specified role weights. Role decomposition makes these weights and the delegated channel explicit, allowing DCE to adjust user-side credit separately.

Proposition 2 (endpoint invariance). If two trajectories end with identical values on every field in $\mathcal{R}_s$, they have the same Φ_s, regardless of their action sequences. This is invariance to the path taken toward the canonical state target, not a guarantee that distinct valid terminal states receive identical potential.

The construction has several limitations. First, fields that agree between x_0 and x^* are excluded, so corrupting a target-invariant field is not penalized by the dense potential. Second, positional flattening treats a reordering of a semantically unordered list as a state change. Third, the potential does not score communication constraints. Equal weighting of flattened fields is also representation-dependent: jointly changing paths remain separate contributions. These conditions motivate retaining the terminal GRPO objective and describing the potential as a partial approximation rather than a decomposition of the evaluator.

B Reward-to-Advantage Implementation

B.1 Operational Cost and Turn Attachment

Operational cost is computed from consecutive verifier snapshots for each assistant turn. The assistant tool-error count is differenced across snapshots; an invalid-action indicator is set when the returned observation reports an invalid action or an action-parsing error; and new transfer calls are penalized only when the task's golden actions contain no transfer. The applicable terms are combined as in Equation 6.

The immediate process reward and operational cost are stored on the final trainable token of the corresponding assistant turn. This records the turn-level signals; the tokens receiving auxiliary supervision are determined by the routing below.

B.2 Return Construction and Token Routing

The backward return in Equation 10 includes the current process reward and discounted rewards from all subsequent assistant turns. Operational cost remains on the turn in which it occurs and is not accumulated backward.

Under the default routing, both normalized channels follow the token-assignment rule described after Equation 13: tool-call and communication tokens receive auxiliary supervision, while reasoning tokens retain only terminal supervision. User, environment, and tool-response tokens are excluded from optimization.

B.3 Centering, Normalization, and Gating

The terminal component is the standard GRPO group-relative advantage computed from binary terminal rewards. For each complete sibling group g, let $n_{i,t}$ count the trainable tool-call and communication tokens in turn t of trajectory i under the default routing. A single baseline is shared across all turns and trajectories in the group:

$$b_g = \frac{\sum_{i \in g} \sum_t n_{i,t} G^p_{i,t}}{\sum_{i \in g} \sum_t n_{i,t}}. \tag{18}$$

We set $b_g = 0$ if the group has no credited tokens. The centered process return is $\bar{G}^p_{i,t} = q_g(G^p_{i,t} - b_g)$, where $q_g = 0$ for all-success groups and $q_g = 1$ for mixed and all-failure groups. This gate is applied before RMS normalization and affects only the process channel; risk penalties remain active. Incomplete or ungrouped samples receive no process contribution.

Let $\mathcal{C}$ contain the assistant turns from complete sibling groups in the batch. When these turns contain credited tokens, the squared process RMS is:

$$\mathrm{RMS}(\bar{G}^p)^2 = \frac{\sum_{(i,t) \in \mathcal{C}} n_{i,t} (\bar{G}^p_{i,t})^2}{\sum_{(i,t) \in \mathcal{C}} n_{i,t}}. \tag{19}$$

Credited tokens in all-success groups remain in the denominator with zero gated returns. Tokens outside the routing mask and turns from incomplete groups are excluded. If there are no credited tokens in complete groups, no token receives a process contribution.

Risk uses a separate RMS over all assistant turns $\mathcal{T}_\mathcal{B}$ in the batch, including zero-cost turns:

$$\mathrm{RMS}(c^{\mathrm{risk}})^2 = \frac{1}{|\mathcal{T}_\mathcal{B}|} \sum_{(i,t) \in \mathcal{T}_\mathcal{B}} (c^{\mathrm{risk}}_{i,t})^2. \tag{20}$$

Unlike the process RMS, this statistic weights turns equally rather than by credited-token count. Risk costs are not group-centered. The two channels are normalized and clipped as in Equations 11 and 12, then combined with terminal supervision through Equation 13. Numerical settings are reported in Appendix E.3.

C Offline Field-DCE Estimation and Diagnostics

C.1 Estimand and Assumptions

For an eligible field f, let $Y_f(A), Y_f(B) \in \{0, 1\}$ indicate whether the field is on target at the end of the complete user turn under the instruction-preserving and neutral interventions. The probabilities:

$$P_A(f) = \Pr(Y_f(A) = 1)\,, \qquad P_B(f) = \Pr(Y_f(B) = 1)$$

marginalize over eligible replay contexts and user-simulator randomness. The untruncated population coefficient is:

$$w_f^\star = \frac{P_A(f) - P_B(f)}{P_A(f)}, \qquad P_A(f) > 0. \tag{21}$$

Under consistency, positivity, and monotonicity $Y_f(A) \geq Y_f(B)$, this quantity admits the probability-of-necessity interpretation (Tian & Pearl, 2000):

$$w_f^\star = \Pr(Y_f(B) = 0 \mid Y_f(A) = 1)\,.$$

Without monotonicity, we use it only as normalized intervention-relative lift. It is a population-level coefficient and does not establish that an individual transition was caused by the instruction. Clipping each coefficient to $[0, 1]$ preserves the sign of each factual field transition or sets it to zero, while discarding estimated negative lift. Because the fixed acknowledgement replaces the entire current instruction-bearing assistant message rather than isolating a token-level instruction, all attribution claims are conditional on this intervention and the selected user simulator.

C.2 Offline Replay Protocol

The reported offline replay set contains 1,673 user turns following instruction-bearing assistant messages from 94 training tasks; evaluation-test tasks are excluded. For each turn, both branches restore the same dialogue prefix and pre-user environment state. Branch A retains the original assistant message, whereas branch B substitutes a fixed non-instructional acknowledgement; the user simulator, decoding configuration, and stopping rule are held fixed. Each branch uses $R = 8$ replays with the same 35B user simulator at temperature 1.0. A replay continues through the complete user turn, including multi-step check–act–verify behavior, for at most 40 user-side interaction steps.

For each field f, the calibration set $\mathcal{I}_f$ consists of recorded turns whose current assistant instruction targets that field. Field selection is based on the instruction, before observing replay outcomes, and both branches use the same contexts. Let $Y_{j,f,r}(b)$ indicate whether f is on target at the end of replay r for turn j under branch $b \in \{A, B\}$. We estimate:

$$\widehat{P}_b(f) = \frac{1}{|\mathcal{I}_f| R} \sum_{j \in \mathcal{I}_f} \sum_{r=1}^{R} Y_{j,f,r}(b), \qquad b \in \{A, B\}. \tag{22}$$

The corresponding raw and stored empirical coefficients are:

$$w_f^{\mathrm{raw}} = \frac{\widehat{P}_A(f) - \widehat{P}_B(f)}{\widehat{P}_A(f)}, \qquad w_f^{\mathrm{off}} = \mathrm{clip}_{[0,1]}\big(w_f^{\mathrm{raw}}\big)\,. \tag{23}$$

The same field-level discount is reused for regressions. A field coefficient is defined when $|\mathcal{I}_f| > 0$ and $\widehat{P}_A(f) > 0$; all 19 runtime keys in the reported calibration set satisfy these conditions. When clipping is inactive, $\widehat{P}_A(f) w_f^{\mathrm{raw}} = \widehat{P}_A(f) - \widehat{P}_B(f)$ over these replay contexts. Thus, the coefficient expresses the relative decrease in target attainment when the current instruction-bearing message is replaced.

C.3 Offline Field-Weight Estimates

Table 4 reports the expanded calibration results used for the frozen field weights. The 19 runtime keys are shown in 15 rows because five coupled VPN fields have identical statistics. Across fields, neutral target-attainment frequencies range from 0.013 to 0.190, while retained-credit coefficients range from 0.772 to 0.973. The variation shows that the neutral intervention affects fields

Field or field family	Calibration turns	$\widehat{P}_B(f)$	w_f^{off}
Airplane mode	361	0.190	0.772
VPN connection (five coupled keys)	126	0.132	0.806
Network connection status	294	0.155	0.808
Data enabled	278	0.154	0.810
Roaming enabled	302	0.125	0.832
Data-saver mode	149	0.116	0.843
SIM-card missing	185	0.132	0.845
Network signal strength	327	0.103	0.854
Connected network technology	315	0.100	0.856
Preferred network mode	394	0.091	0.875
App status: storage	153	0.064	0.888
Wi-Fi calling enabled	121	0.050	0.933
App status: SMS	129	0.043	0.934
Active APN: MMSC URL	79	0.034	0.944
Active APN: APN name	30	0.013	0.973

Table 4: Offline Field-DCE calibration over 1,673 turns from 94 training tasks. Turn counts are field-specific and can overlap. $\widehat{P}_B(f)$ is neutral target-attainment frequency; w_f^{off} is the frozen coefficient computed from both branches, not $1 - \widehat{P}_B(f)$. Five coupled VPN keys are displayed together but remain separate fields in the reward. Values are rounded to three decimals.

differently: airplane-mode progress receives stronger attenuation than APN configuration. Offline DCE therefore preserves most factual credit while applying a field-specific correction derived from instruction-preserving and neutral replay.

These coefficients summarize the calibration-context distribution. They expose what Offline Field-DCE retains and attenuates, but cannot adapt to differences between turns involving the same field. The contextual diagnostics in Appendix D.7 directly test the value of such within-field variation by matching the removal budget separately for each field and training stage.

D Contextual DCE Replay and Estimator Properties

D.1 Local Full-Turn Replay

With the zero-change channel disabled, Online Contextual DCE triggers replay when at least one user-side target field changes its target-match status and a preceding non-empty assistant message is available. The trigger uses observed state changes, without a separate classifier for instruction-bearing messages. The observed rollout is factual branch A and is not resampled. For branch B, we restore the preceding dialogue and pre-user environment state, replace the latest preceding non-empty assistant message with the fixed control, and replay the complete ensuing user turn with fresh simulator randomness, including its tool chain and synchronized responses. No later trajectory suffix is replayed. Neutral samples yield field-specific attainment and regression frequencies, $\widehat{p}^{+}$ and $\widehat{p}^{-}$. This local construction measures recurrence at the boundary where delegated progress is scored.

D.2 Neutral Intervention

The fixed control is the non-interrogative acknowledgement *Understood. I've noted the information you provided.* It contains no question, action name, next-step suggestion, or invitation to continue. The intervention replaces the whole assistant message, including its instructions, factual information, and tone; it does not isolate executable requests alone. The preceding dialogue and pre-user state are retained, so information from earlier exchanges can still influence the user simulator. The estimand is therefore conditional on this message replacement, the retained context, and the selected simulator.

A more selective intervention could remove only executable requests while retaining the rest of the message. We use a fixed acknowledgement to avoid an additional learned or rule-based masking component that could itself alter the message's meaning.

D.3 Bounded Contextual Integration

For neutral replay ℓ, let $z_{t,f}^{B,\ell} \in \{-1, 0, +1\}$ follow the same target-relative rule as the factual transition $z_{t,f}^{A}$ in Equation 7. From n matched neutral replays with fresh simulator randomness, we form empirical progress and regression frequencies:

$$\widehat{p}_{t,f}^{+} = \frac{1}{n}\sum_{\ell=1}^{n} \mathbf{1}\left\{z_{t,f}^{B,\ell} = +1\right\}, \qquad \widehat{p_{t,f}^{-}} = \frac{1}{n}\sum_{\ell=1}^{n} \mathbf{1}\left\{z_{t,f}^{B,\ell} = -1\right\}. \tag{24}$$

In the reported Online Contextual DCE experiment, we use $n = 8$ neutral replays per candidate and $\lambda_0 = 0$. For nonzero factual changes, the estimator applies a nonnegative discount to observed progress or regression. Let $0 \le \lambda_0 \le 1$ control optional opportunity credit when the factual field is unchanged:

$$\widetilde{g}_{t,f} = \begin{cases} 1 - \widehat{p}_{t,f}^{+}, & z_{t,f}^{A} = +1, \\ \lambda_0(\widehat{p_{t,f}^{-}} - \widehat{p}_{t,f}^{+}), & z_{t,f}^{A} = 0, \\ -(1 - \widehat{p_{t,f}^{-}}), & z_{t,f}^{A} = -1, \end{cases} \qquad g_t^{\mathrm{del,on}} = \frac{1}{|\mathcal{R}_u|}\sum_{f \in \mathcal{R}_u} \widetilde{g}_{t,f}. \tag{25}$$

Proposition 3 (per-field bound). For $0 \le \widehat{p}_{t,f}^{+}, \widehat{p_{t,f}^{-}}, \lambda_0 \le 1$, the per-field score in Equation 25 satisfies:

$$-1 \le \widetilde{g}_{t,f} \le 1. \tag{26}$$

Proof. For factual progress and regression, the multipliers $1 - \widehat{p}^{+}$ and $1 - \widehat{p}^{-}$ lie in $[0, 1]$. For factual no-change, $\widehat{p}^{-} - \widehat{p}^{+} \in [-1, 1]$, and multiplying by $\lambda_0 \in [0, 1]$ preserves the interval. □

When $\widehat{p}^{+} = \widehat{p}^{-} = 0$ and factual transitions are non-zero, the contextual rule reduces to factual per-field potential. With $\lambda_0 = 0$, factually unchanged fields contribute zero regardless of neutral replay.

Statistical target and expected credit. Fix a dialogue context, pre-user state, field, simulator, and neutral message. For an initially off-target field with observed factual attainment $z_{t,f}^{A} = +1$, let p_B be its attainment probability in an independently sampled neutral user turn under the stated replay stopping rule. The retained-credit weight $\widehat{w} = 1 - \widehat{p}_{t,f}^{+}$ estimates the probability that this observed gain does not recur under neutral replay. For n independent identically distributed neutral samples:

$$\mathbb{E}[\widehat{w} \mid z_{t,f}^{A} = +1] = 1 - p_B, \qquad \mathrm{Var}(\widehat{w} \mid z_{t,f}^{A} = +1) = \frac{p_B(1 - p_B)}{n}. \tag{27}$$

The fixed context is implicit in both conditional expressions. The same result holds for observed regressions with p_B replaced by the neutral regression probability. Additional replays reduce sampling variance while preserving the statistical target.

For the reported setting $\lambda_0 = 0$, let p_A denote the factual attainment probability. If factual outcomes are also sampled independently of the neutral replays, then $\mathbb{E}[\widetilde{g}_{t,f}] = p_A(1 - p_B)$. This is recurrence-adjusted factual credit, rather than the mean intervention effect $p_A - p_B$: for example, $p_A = p_B = 0.5$ gives expected credit 0.25. Setting $\lambda_0 = 0$ deliberately restricts correction to observed changes. It assigns no opportunity credit to unchanged factual fields, even when a neutral branch would have changed them. The diagnostics below test whether estimated recurrence generalizes to disjoint replay samples and directs credit removal beyond constant attenuation.

For a single neutral replay, Table 5 shows the field contribution $u_f \widetilde{g}_{t,f}$, where $u_f = 1/|\mathcal{R}_u|$. Opposite-sign changes are infeasible because both branches start from the same binary target-match state.

D.4 Interpretation and Scope

The estimator scores each field at the end of the complete user turn, capturing the net target-status change across multi-step check–act–verify chains. Matching only the first user tool call would miss changes produced later in the turn. Applying the correction before aggregating fields also distinguishes target attainment and regression that could cancel in the total device potential.

The scores adjust factual credit under the retained dialogue, selected simulator, and fixed acknowledgement. Appendices D.6 and D.7 validate their ability to identify and target neutrally reproducible progress using disjoint replay samples. Section 5.3 separately evaluates the resulting policies.

Factual z^A \ Neutral z^B	$+1$	0	-1
$+1$	0	$+u_f$	—
0	$-\lambda_0 u_f$	0	$+\lambda_0 u_f$
-1	—	$-u_f$	0

Table 5: Per-field contributions $u_f \widetilde{g}_{t,f}$ under a single neutral replay for the generalized correction, where $u_f = 1/|\mathcal{R}_u|$. Reported Online Contextual DCE uses eight neutral replays and $\lambda_0 = 0$. Dashes mark transition pairs incompatible with the same initial target-match status.

Cancellation with fixed and contextual weights. Fixed Offline field weights define a weighted state potential and preserve raw, undiscounted telescoping. Online weights vary with context and need not preserve it. Consider a field that reaches its target and then returns to its original value. If the corresponding neutral attainment and regression frequencies are 0 and 1, the two Online field contributions are $+1$ and 0 before field averaging. Their sum is nonzero despite identical initial and final field values. This is a limitation of credit cancellation, not evidence that the trained policy exploits such cycles.

D.5 Replay Coverage, Cost, and Regression Audit

We audit the recorded full-strength replay phase, covering 11,032 trajectories from 1,299 tasks. Each of the 15,273 replay candidates has eight usable neutral outcomes, totaling 122,184 user-turn replay attempts. No execution exceptions or all-replays-failed fallbacks were recorded. Of these attempts, 121,555 ended normally and 629 reached the 15-step replay limit (0.51%). Step-limited outcomes remain usable: field changes are measured at their final observed states, consistently with training. Table 6 summarizes the audit.

Audit quantity	Value
Recorded trajectories / tasks	11,032 / 1,299
Replay candidates	15,273
Neutral user-turn replay attempts	122,184
Normal termination	121,555
Step-limit termination	629 (0.51%)
Execution exceptions	0 / 122,184
All-replays-failed fallback	0 / 15,273
Positive field events / tasks	59,294 / 1,283
Regression field events / tasks	72 / 31
Positive factual-credit mass retained	93.8%
Absolute regression-credit mass retained	90.5%

Table 6: Replay and credit audit for the recorded full-strength Online DCE phase. Usable outcomes include step-limited replays; absence of execution exceptions does not imply task success. Credit retention is measured before advantage construction.

Replay cost. The observed workload averages 11.08 neutral user-turn replays per recorded trajectory, with replay concentrated on candidate turns. These are user-turn simulations, each potentially requiring multiple simulator calls; they are not API-call counts. The retained logs do not support a corresponding token-usage or wall-clock estimate.

Regression coverage. Online DCE retains 93.8% of positive factual-credit mass and 90.5% of absolute regression-credit mass. Thus, the runtime audit covers both signs of state change. The held-out identification and targeting diagnostics use the substantially larger population of positive events.

Closed-cycle residual credit. We identify 72 non-overlapping target-attainment–reversal cycles across 33 trajectories and 31 tasks. Five yield positive residual DCE credit, totaling 0.111 in field-normalized units, or approximately 0.0015% of the 7,536.1 total positive DCE credit mass. Although contextual weights need not preserve exact cancellation, positive residual credit from these observed cycles is negligible relative to the positive-credit mass in this audit. These measurements concern

detected cycles and raw credit, before advantage normalization; they do not quantify unobserved cycles or their effects on policy updates.

D.6 Held-out Identification of Neutral-Reproducible Progress

We examine whether contextual replay identifies observed gains that recur under the neutral intervention. The analysis uses 59,294 positive field–turn events from 1,283 tasks in the recorded Online Contextual DCE training trajectories during the full-strength replay phase. Each event is a factual transition from an off-target value to its target. Policy training uses all eight neutral outcomes; for this diagnostic, we partition them into two disjoint sets of four using seed 42. All results reuse recorded replays, with no additional simulator calls.

Let $\mathcal{J}$ be the set of positive events and let $q_j^{(1)}$ and $q_j^{(2)}$ be the fractions of neutral replays reproducing event j in the first and second halves. The first half selects events; the second evaluates their neutral reproducibility. For threshold τ, define:

$$\mathcal{F}_\tau = \{j \in \mathcal{J} : q_j^{(1)} \geq \tau\}, \qquad \text{Coverage}(\tau) = \frac{|\mathcal{F}_\tau|}{|\mathcal{J}|}, \qquad \text{Recall}(\tau) = \frac{\sum_{j\in\mathcal{F}_\tau} q_j^{(2)}}{\sum_{j\in\mathcal{J}} q_j^{(2)}}. \tag{28}$$

Events receive equal weight in this identification analysis. Since every event has four validation replays, recall is also the fraction of all held-out positive replay outcomes covered by the selected events. Held-out refers to replay samples at the same recorded contexts, rather than additional benchmark tasks.

Selection threshold τ	Selected events (%)	Held-out recall [95% CI]	Selected recurrence
0.25	13.62	0.7557 [0.740, 0.771]	0.3365
0.50	6.66	0.5155 [0.495, 0.537]	0.4694
0.75	2.98	0.2653 [0.247, 0.285]	0.5410
1.00	0.77	0.0754 [0.066, 0.086]	0.5922

Table 7: Identification using four neutral replays and validation using the other four. Selected recurrence is the mean $q_j^{(2)}$ among selected events. Confidence intervals use task-cluster bootstrap resampling (Field & Welsh, 2007).

At $\tau = 0.25$, only 13.62% of events account for 75.57% of neutral recurrences in the validation half. Their held-out recurrence frequency is 0.3365 (95% CI: $[0.324, 0.349]$), versus approximately 0.017 ($[0.016, 0.018]$) for unselected events. Increasing the threshold selects fewer events with progressively higher held-out recurrence. The large separation across disjoint replay samples demonstrates reproducible contextual variation, rather than merely selecting fluctuations in the samples used to compute a correction. The 13.62% selection rate describes this four-replay diagnostic; it is distinct from the 18.3% of positive events attenuated using all eight replays during training.

D.7 Credit Targeting under Matched Removal Budgets

The identification result treats all events equally. We next test whether contextual DCE allocates a fixed amount of removed credit more selectively than constant attenuation. For positive event j, let $g_j > 0$ be its raw factual delegated contribution, including the field normalization $1/|\mathcal{R}_u|$. The contextual removal and its total budget are:

$$d_j^{\text{ctx}} = g_j q_j^{(1)}, \qquad D = \sum_{j\in\mathcal{J}} d_j^{\text{ctx}}. \tag{29}$$

A global constant removes $d_j^{\text{global}} = \kappa g_j$, where $\kappa = D/\sum_j g_j$. For the stronger control, partition events into strata s defined by field and training stage. The early, middle, and late stages comprise recorded rollout indices 0–39, 40–79, and 80–119. With $G_s = \sum_{j\in s} g_j$ and $D_s = \sum_{j\in s} d_j^{\text{ctx}}$, field–stage attenuation removes:

$$d_j^{\text{fs}} = g_j \frac{D_s}{G_s}, \qquad j \in s. \tag{30}$$

Thus, all three rules remove exactly D credit, and the field–stage control also preserves every stratum's removal budget. On the recorded events, $D = 499.691$ in normalized field-credit units and $\kappa \simeq 0.0622$. The controls differ only in how this fixed budget is allocated across events.

We evaluate targeting using the second replay half:

$$S(d) = \frac{\sum_{j \in \mathcal{J}} d_j q_j^{(2)}}{\sum_{j \in \mathcal{J}} d_j}. \tag{31}$$

This is the neutral recurrence frequency averaged with weights proportional to removed credit. A higher score means that removal is concentrated on factual gains that recur more often under the neutral intervention. It does not classify individual gains as deterministically instruction-independent.

Credit-removal rule	Targeting score S [95% CI]	Paired gain of contextual DCE
Global constant	0.0626 [0.059, 0.067]	0.3467 [0.335, 0.358]
Field–stage constants	0.0808 [0.075, 0.089]	0.3285 [0.316, 0.340]
Contextual DCE	**0.4093** [0.396, 0.423]	—

Table 8: Targeting at an identical credit-removal budget. Weights use the first four neutral replays and scores use the remaining four. Paired gains subtract each control's score from contextual DCE. Intervals use 2,000 task-cluster bootstrap replicates, retaining all events and replay outcomes of a sampled task and comparing rules within the same replicate.

Contextual DCE achieves $S = 0.4093$, compared with 0.0626 for a global constant and 0.0808 for field–stage constants (Table 8). The paired differences are positive throughout their 95% intervals. Moreover, events with held-out recurrence at least 0.5 receive 52.2% of contextual removal, versus 6.9% under global attenuation. Hence, the advantage is not explained by a larger removal budget or by assigning larger discounts to particular fields or training stages.

The comparison with the field–stage control has a direct algebraic interpretation. Define $\mathrm{Cov}_{g,s}$ as the empirical covariance within stratum s, with normalized weights g_j/G_s. Then:

$$D\big[S(d^{\mathrm{ctx}}) - S(d^{\mathrm{fs}})\big] = \sum_s G_s \,\mathrm{Cov}_{g,s}\Big(q^{(1)}, q^{(2)}\Big). \tag{32}$$

The positive paired difference therefore measures cross-half reproducibility within field–stage strata. Together, the split-replay and matched-budget results establish that contextual DCE captures useful information beyond field and stage averages and uses it to target attenuation toward neutrally reproducible progress. These controls evaluate credit allocation on recorded trajectories; they are not additional trained policy variants. Policy-level results are reported separately in Section 5.3.

E Experimental Details

E.1 Environment, Tasks, and Data Sources

We use a team-maintained fork of τ^2-bench-Verified, version `0.2.1-dev`, commit `864350a` (2026-04-02). We retain the available Airline data from AReaL and synthesize data for Retail and Telecom to obtain roughly balanced training-set sizes across domains. All RL variants are jointly trained on the same merged collection. Table 9 summarizes the training sources and evaluation splits, distinguishing tasks from the trajectories produced by five trials per task.

Domain	Training source	Training instances	Evaluation tasks	Evaluation trajectories/run
Airline	AReaL data (Gao et al., 2026)	1,148	50	250
Retail	563 existing + 615 synthetic	1,178	114	570
Telecom	Synthetic tasks	1,300	114	570
Total		3,626	278	1,390

Table 9: Training instances, distinct base-split evaluation tasks, and evaluation trajectories per run. Each evaluation run contains five trials per task. Our synthesis pipeline adds no Airline data.

Configuration	Process signal	Auxiliary tokens	Gate	Risk
Terminal GRPO	None	None	—	No
ToolRL (adapted)	Call matching	All assistant tokens	Off	No
Direct-only	Direct state progress	Tool calls and communication	On	Yes
Direct + Delegated	Direct + factual delegated	Tool calls and communication	On	Yes
Offline / Online DCE	Direct + adjusted delegated	Tool calls and communication	On	Yes
All tokens ablation	Direct + factual delegated	All assistant tokens	On	Yes

Table 10: Configuration differences relevant to the reported comparisons. Gate denotes the all-success process gate. Terminal credit reaches all trainable assistant tokens in every RL configuration; auxiliary routing applies to process and risk signals when present. All configurations use 120 training steps, which does not imply equal computation.

Retail synthesis. We augment the existing Retail data with instances synthesized by `Qwen3.5-397B-A17B-FP8` from failed trajectories collected during AReaL RL training. The source trajectories come from Retail training tasks that are disjoint from the evaluation tasks.

Telecom synthesis. The available Telecom training data share customer information with the evaluation set. We therefore use `DeepSeek-V4-Pro` to generate three new customer profiles from scratch. These profiles are used to instantiate tasks drawn from the official `full` task collection after excluding the `base` evaluation split. We then sample the resulting instances for RL training.

DCE replay contexts are drawn only from training tasks.

E.2 Models and Compared Systems

All RL variants initialize from the official post-trained `Qwen/Qwen3.5-9B` checkpoint. We use `Qwen3.6-35B-A3B` as the fixed user simulator with the benchmark's scenario prompts. During both RL training and evaluation, the user simulator uses a temperature of 1.0 with thinking enabled. Base uses the initial checkpoint without additional RL, and Terminal GRPO uses only terminal task-success rewards. The state-grounded variants follow Appendices A and B; DCE changes only the delegated channel as specified in Appendices C and D.

ToolRL reward. ToolRL (adapted) (Qian et al., 2025) combines a format reward $R_{i,t}^{\text{format}}$ with a correctness reward $R_{i,t}^{\text{correct}}$. The format component checks whether the assistant response can be parsed by the environment; the correctness component scores agreement with reference tool calls through tool-name, parameter-name, and parameter-value matching.

A successfully parsed interaction step containing tool calls from either role receives $R_{i,t}^{\text{format}} = 1$, whereas an assistant-response parse failure receives $R_{i,t}^{\text{format}} = 0$. Live scoring uses the environment's parse result. Offline scoring additionally checks that response tags are balanced when raw assistant text is available; this check is not applied to live messages that already separate reasoning, content, and tool calls. Format validity is scored separately from agreement with the reference actions.

For assistant turn t in trajectory i, let $P_{i,t}$ contain all tool calls executed by the assistant and user simulator in the corresponding interaction step. Let $\mathcal{G}_i$ be the task's full ordered list of golden actions, including actions assigned to both roles. For a valid step containing tool calls, name coverage is:

$$s_{i,t}^{\text{name}} = \frac{1}{|P_{i,t}|} \sum_{p \in P_{i,t}} \mathbf{1}\{\text{name}(p) \in \text{names}(\mathcal{G}_i)\}\,. \tag{33}$$

Golden actions are visited in their declared order and greedily paired with the highest-scoring unused same-name call. The pair score $a(p, g)$ is the Jaccard similarity of parameter-name sets plus the number of exactly matching parameter values, comparing full arguments. The Jaccard similarity of two empty sets is 1. Let $\mathcal{M}_{i,t}$ be the resulting pairs. Let $R_{\max} > 0$ denote the fixed magnitude bound of the correctness reward. We set $R_{\max} = 3$ to retain ToolRL's original correctness-reward

range (Qian et al., 2025), mapping the normalized matching score $s_{i,t}/S^{\max}_{i,t}$ linearly from $[0,1]$ to $[-R_{\max}, R_{\max}]$:

$$s_{i,t} = s^{\text{name}}_{i,t} + \sum_{(p,g)\in\mathcal{M}_{i,t}} a(p,g), \tag{34}$$

$$S^{\max}_{i,t} = 1 + \sum_{(p,g)\in\mathcal{M}_{i,t}} \big(1 + |\operatorname{params}(g)|\big), \tag{35}$$

$$R^{\text{correct}}_{i,t} = R_{\max}\big(2\, s_{i,t}/S^{\max}_{i,t} - 1\big), \qquad r^{\text{TL}}_{i,t} = R^{\text{format}}_{i,t} + R^{\text{correct}}_{i,t}. \tag{36}$$

Only matched golden actions enter $S^{\max}_{i,t}$, avoiding dilution by unmatched task-level references. Each predicted call is used at most once per step. Matching restarts at every step, so repeated golden actions can earn repeated rewards and cross-turn execution order is not enforced.

Under this mapping, a parseable tool-call step with no name match receives the minimum correctness reward $R^{\text{correct}}_{i,t} = -R_{\max} = -3$ and total reward $1 - 3 = -2$. An assistant-response parse failure is also assigned this minimum correctness reward; together with its zero format reward, this gives total reward -3. A valid step with no tool calls from either role is not scored and receives $r^{\text{TL}}_{i,t} = 0$, so dialogue-only continuation earns no format bonus. The parse-failure rule takes precedence over this no-call exception.

Return and routing. The immediate reward is attached to the final trainable token of the assistant step. With $\gamma = 1$, the process return is $G^{\text{TL}}_{i,t} = \sum_{k\geq t} r^{\text{TL}}_{i,k}$. We use the token-weighted group baseline in Equation 18, with $n_{i,t}$ counting all trainable assistant tokens, including reasoning. Centered returns undergo the batch RMS normalization of Appendix B and clipping to $[-3,3]$, yielding $\widehat{G}^{\text{TL}}_{i,t}$. Combined with the same terminal GRPO advantage used in the main experiments, the update uses:

$$A^{\text{TL}}_{i,t} = A^{\text{term}}_{i} + 0.2\,\widehat{G}^{\text{TL}}_{i,t}. \tag{37}$$

The process gate is disabled and $\beta_r = 0$. Both terminal and process credit reach all trainable assistant tokens; user-simulator and environment tokens are never optimized. Consequently, call-free steps can receive future matching credit despite zero immediate reward.

These choices adapt ToolRL's reward decomposition to task-level references without turn alignment. Valid alternative calls absent from the reference list can receive negative rewards, while matching reference calls does not guarantee terminal success. The baseline therefore evaluates this adapted reward together with the specified return, routing, and gating settings.

Neither the executed calls nor the golden actions are filtered by role: $P_{i,t}$ pools assistant and user-simulator calls, and $\mathcal{G}_i$ retains user-side reference actions alongside assistant-side actions. Matching uses tool names and arguments without treating requestor identity as a matching criterion. Calls from both roles enter the name-coverage denominator, and matched golden actions from either role contribute to $S^{\max}_{i,t}$. Thus, an assistant message followed by user tool calls can receive process reward through those calls even when the assistant makes no tool call itself. The baseline aggregates these contributions into one action-matching reward rather than separate direct and delegated channels. This reward trains only the assistant policy; it does not optimize the user simulator.

E.3 Training Configuration and Hyperparameters

Table 11 summarizes the training, evaluation deployment, and state-grounded reward settings. All RL variants are evaluated after 120 training steps (zero-based step 119), corresponding to approximately one epoch through the training data. The reward-to-advantage construction follows Appendix B; ToolRL-specific settings are given in Appendix E.2. Each RL variant is trained once, using a global random seed of 1234 and a rollout-shuffling seed of 42. The three evaluation repeats reported below evaluate the resulting checkpoint rather than independently trained models.

Each assistant response is capped at 4,096 generated tokens; responses reaching this limit are truncated without retry. Training trajectories are limited to 200 steps and 50 assistant turns. Trajectories reaching the cumulative response budget of 32,768 tokens are truncated and retained for training.

Ablation settings. The routing ablation applies both process and risk credit either to tool-call and communication tokens (Routed) or to all trainable assistant tokens, including reasoning (All tokens),

Setting	Value
Training samples per prompt	8
Rollout / global batch size	32 / 256
Optimizer / weight decay	Adam (Kingma & Ba, 2015) / 0.1
Learning rate / schedule	10^{-6} / cosine
KL coefficient	0.001
Policy-ratio clipping (lower / upper)	0.2 / 0.28
Importance sampling	Token level
Assistant rollout temperature	1.0
Maximum generated tokens per assistant response	4,096
Cumulative response-token budget per trajectory	32,768
Maximum steps / assistant turns per trajectory	200 / 50
Precision / parallelism	bf16; TP2 / CP4; fully asynchronous
State-grounded reward settings	
Process-return discount γ	1.0
Factual-delegated role weights $(\alpha_{\mathrm{dir}}, \alpha_{\mathrm{del}})$	$(1.0, 1.0)$
Direct-only role weights $(\alpha_{\mathrm{dir}}, \alpha_{\mathrm{del}})$	$(1.0, 0)$
Common immediate progress scale	1.0
Operational costs $(\lambda_{\mathrm{err}}, \lambda_{\mathrm{inv}}, \lambda_{\mathrm{tr}})$	$(0.10, 0.10, 0.15)$
RMS stabilizer $\varepsilon_{\mathrm{rms}}$	10^{-8}
Auxiliary clipping threshold c	3
Auxiliary strengths (β_p, β_r)	$(0.20, 0.10)$
Evaluation deployment settings	
Assistant context window	131,072 tokens (128k)
User-simulator context window	65,536 tokens (64k)
Maximum evaluation steps (Airline / Retail / Telecom)	50 / 50 / 150

Table 11: Training and evaluation configurations, including default auxiliary-reward settings for the state-grounded variants. Role weights are fixed by variant rather than tuned independently.

with $(\beta_p, \beta_r) = (0.2, 0.1)$ in both variants. The process-strength ablation in Section 5.4 uses factual delegated progress with $\beta_p \in \{0, 0.2, 0.5\}$, fixing $\beta_r = 0.1$ and retaining the default routing and remaining settings.

DCE configuration. Offline DCE uses frozen field-level credit weights estimated before policy training, as described in Appendix C, with eight replays per branch and a 40-step limit. Online DCE uses eight neutral replays when user-side target-match status changes, following the trigger in Appendix D. Each replay holds the preceding dialogue and pre-user state fixed, replaces the entire selected assistant message with the acknowledgement, and simulates the ensuing user turn at temperature 1.0 with thinking enabled and a 15-step limit. Samples with execution failures are excluded; if no usable neutral sample remains, factual delegated credit is retained without an additional penalty. Step-limited samples use their final observed field states. Appendix D.5 reports the observed replay workload and completion statistics. Offline and Online DCE differ in replay budget, estimation granularity, and context source.

Estimator coverage, field weights, and replay controls are reported in Appendices C and D.

E.4 Evaluation and Statistical Protocol

Each evaluation run uses five trials per task on the base splits in Table 9. Evaluation step limits and deployed context windows are listed in Table 11. No explicit per-response output-token limit is set. Success is measured by the unmodified τ^2-bench-Verified terminal evaluator; process and risk rewards are used only in training. For run e, let $c_{j,e}$ count successful trials for task j in the domain-specific task set $\mathcal{T}$. We compute:

$$\widehat{\mathrm{Pass}}_e^{\,k} = \frac{1}{|\mathcal{T}|} \sum_{j \in \mathcal{T}} \frac{\binom{c_{j,e}}{k}}{\binom{5}{k}}, \qquad k \in \{1, \ldots, 5\}, \tag{38}$$

Method	Pass^1	Pass^2	Pass^3	Pass^4	Pass^5
Airline					
Qwen3.5-9B Base	74.0 ± 0.7	64.4 ± 0.2	58.4 ± 0.2	54.3 ± 0.6	51.3 ± 1.2
Terminal GRPO	75.9 ± 2.3	68.5 ± 2.5	63.7 ± 2.5	60.1 ± 2.6	57.3 ± 3.1
ToolRL (adapted)	72.5 ± 0.5	64.3 ± 1.3	59.3 ± 1.9	55.9 ± 2.4	53.3 ± 3.1
Direct-only	75.2 ± 0.0	67.7 ± 0.6	62.8 ± 1.2	59.2 ± 1.7	56.7 ± 2.3
Direct + Delegated	76.3 ± 0.6	68.7 ± 0.9	63.8 ± 1.8	60.4 ± 2.8	58.0 ± 3.5
Retail					
Qwen3.5-9B Base	76.5 ± 0.5	64.1 ± 0.6	55.9 ± 1.2	49.8 ± 1.8	45.0 ± 2.2
Terminal GRPO	75.9 ± 1.1	63.6 ± 1.6	54.9 ± 2.6	48.4 ± 3.6	43.0 ± 4.6
ToolRL (adapted)	76.9 ± 0.4	64.5 ± 0.3	56.3 ± 0.4	50.4 ± 0.8	45.9 ± 1.4
Direct-only	76.7 ± 0.4	64.8 ± 0.4	57.1 ± 0.3	51.6 ± 0.3	47.7 ± 0.5
Direct + Delegated	80.3 ± 0.9	69.2 ± 1.4	61.6 ± 1.4	55.7 ± 1.2	51.2 ± 1.0
Telecom					
Qwen3.5-9B Base	90.4 ± 0.5	82.8 ± 0.9	76.7 ± 1.0	71.8 ± 1.0	67.8 ± 1.3
Terminal GRPO	94.0 ± 0.6	88.7 ± 0.9	83.8 ± 1.1	79.5 ± 1.2	75.7 ± 1.3
ToolRL (adapted)	94.5 ± 0.2	89.6 ± 0.4	85.1 ± 0.6	81.1 ± 0.8	77.5 ± 1.0
Direct-only	95.4 ± 0.5	91.3 ± 0.9	87.4 ± 1.4	83.9 ± 1.9	80.7 ± 2.3
Direct + Delegated	96.2 ± 0.4	92.6 ± 0.7	89.2 ± 0.8	85.9 ± 1.0	82.7 ± 1.0
Offline Field-DCE	95.7 ± 0.3	92.0 ± 0.4	88.6 ± 0.5	85.8 ± 0.6	83.3 ± 0.9
Online Contextual DCE	96.0 ± 0.7	92.5 ± 0.9	89.5 ± 0.8	87.0 ± 0.6	84.8 ± 0.5

Table 12: Repeat-to-repeat evaluation variability for the primary methods and DCE variants (%). Values are means $\pm$ sample standard deviations over three repeated evaluation runs under the same master-seed configuration ($n = 3$). Pass^1–Pass^5 are computed separately within each evaluation repeat from five trials per task; trials are not pooled across repeats.

Domain	Routing	β_p	Pass^1	Pass^2	Pass^3	Pass^4	Pass^5
Airline	Routed	0.2	76.3 ± 0.6	68.7 ± 0.9	63.8 ± 1.8	60.4 ± 2.8	58.0 ± 3.5
Airline	All tokens	0.2	73.3 ± 1.5	65.1 ± 2.3	60.0 ± 2.9	56.3 ± 3.2	53.3 ± 3.1
Airline	Routed	0	74.0 ± 1.7	66.7 ± 1.7	62.3 ± 1.4	58.9 ± 1.6	56.0 ± 2.0
Airline	Routed	0.5	75.9 ± 1.6	68.6 ± 2.4	64.2 ± 3.3	60.8 ± 4.3	58.0 ± 5.3
Retail	Routed	0.2	80.3 ± 0.9	69.2 ± 1.4	61.6 ± 1.4	55.7 ± 1.2	51.2 ± 1.0
Retail	All tokens	0.2	77.8 ± 1.5	66.6 ± 1.3	58.2 ± 1.1	51.5 ± 1.0	45.9 ± 1.4
Retail	Routed	0	75.2 ± 1.0	63.0 ± 0.9	54.9 ± 1.8	49.3 ± 2.7	45.0 ± 3.4
Retail	Routed	0.5	78.7 ± 0.8	67.1 ± 1.2	59.3 ± 1.0	53.7 ± 0.5	49.7 ± 1.4
Telecom	Routed	0.2	96.2 ± 0.4	92.6 ± 0.7	89.2 ± 0.8	85.9 ± 1.0	82.7 ± 1.0
Telecom	All tokens	0.2	94.6 ± 1.0	90.2 ± 1.4	86.4 ± 1.6	83.1 ± 1.6	80.1 ± 1.8
Telecom	Routed	0	94.8 ± 0.1	90.0 ± 0.3	85.7 ± 0.5	81.7 ± 0.9	78.0 ± 1.5
Telecom	Routed	0.5	96.1 ± 1.1	92.4 ± 1.9	89.0 ± 2.4	85.8 ± 2.8	82.8 ± 3.1

Table 13: Repeat-to-repeat evaluation variability for the routing and process-strength ablations (%). Entries are means $\pm$ sample standard deviations over three evaluation repeats under the same master-seed configuration, with five trials per task in each repeat. Both process and risk credit follow the indicated routing, with $\beta_r = 0.1$ throughout. The Routed, $\beta_p = 0.2$ rows repeat Direct + Delegated from Table 12 to provide a complete comparison within this table.

where $\binom{c_{j,e}}{k} = 0$ for $c_{j,e} < k$. Pass^k estimates the probability that k trials sampled without replacement all succeed; Pass^1 is mean single-trial success, and Pass^5 is the fraction of tasks solved in all five trials.

Decoding and repeated evaluation. The user simulator uses temperature 1.0 with thinking enabled, following Qwen's recommended temperature for general-task thinking mode.[1] Preliminary evaluation-pipeline checks with temperature zero produced output-format failures, so this setting was not adopted; these checks were not a controlled temperature ablation.

Each checkpoint is evaluated three times with master seed 300, shared across runs and methods. The shared master seed did not yield identical end-to-end outputs under stochastic decoding; differences in early responses can propagate through subsequent dialogue and tool-use turns. For each domain,

[1] `https://huggingface.co/Qwen/Qwen3.6-35B-A3B`

we report the mean and sample standard deviation of the separately computed run-level metrics:

$$\overline{\text{Pass}}^{k} = \frac{1}{3}\sum_{e=1}^{3} \widehat{\text{Pass}}_{e}^{\,k}, \tag{39}$$

$$s_k = \sqrt{\frac{1}{n-1}\sum_{e=1}^{n}\left(\widehat{\text{Pass}}_{e}^{\,k} - \overline{\text{Pass}}^{k}\right)^2}, \qquad n = 3. \tag{40}$$

The fifteen trials across runs are not pooled. Standard deviations describe repeat-to-repeat evaluation variability under the same master-seed configuration, not variability across independent training runs or training seeds.

E.5 Evaluation Variability

Table 12 reports Pass[1]–Pass[5] means and sample standard deviations for the primary methods and DCE variants. Table 13 extends this reporting to both ablations in Section 5.4: the All tokens setting applies process and risk credit to all trainable assistant tokens at $\beta_p = 0.2$, while the Routed settings vary process strength with $\beta_p \in \{0, 0.2, 0.5\}$. The Routed, $\beta_p = 0.2$ rows repeat Direct + Delegated from Table 12 to provide a complete comparison within this table. All entries follow the statistical protocol in Appendix E.4; standard deviations describe repeated evaluation of each checkpoint, not variation across independently trained models.