Calibration Is Not Control: Intervention Value for LLM-Agent Oversight

Chubin Zhang[1] **Zhenglin Wan**[2] **Xingrui Yu**[3,4,‡] **Jingxuan Wu**[5]
Qi Wen[2] **Pengfei Zhou**[2] **Wangbo Zhao**[2] **Ivor Tsang**[1,3,4]

[1]Nanyang Technological University, Singapore [2]National University of Singapore, Singapore
[3]CFAR, Agency for Science, Technology and Research, Singapore
[4]IAIC, Agency for Science, Technology and Research, Singapore
[5]Department of Statistics and Operations Research, UNC-Chapel Hill, United States

[‡]Corresponding author: yu_xingrui@a-star.edu.sg

Abstract

Runtime oversight often intervenes when an LLM agent's calibrated failure score crosses a threshold. Yet states with the same failure risk can differ in whether intervention helps. Strictly increasing recalibration preserves the threshold policy class and cannot recover this distinction. We formalize when a summary is sufficient for intervention decisions and the utility lost when it is not. We evaluate the consequences by replaying agent prefixes and executing alternative actions from the same state. On ALFWorld, holding features, estimator, and router fixed while changing the supervision target from failure to intervention utility lowers regret from 0.51 to 0.09; the gain replicates on a second suite of mid-episode prefixes. A deployable intervention-trained scalar also beats the failure-score threshold rule selected on test outcomes. Online, on 300 unseen tasks with a fixed stronger-model handoff, a frozen prefix-feature controller improves utility over failure-triggered routing, handing off less often (35% vs 48%) and succeeding more often (45% vs 37%). Gains depend on intervention value and are small on two reasoning benchmarks. Oversight signals should be evaluated by the decisions they support alongside their predictive quality. Code is available at `https://github.com/bennidict23/calibration-is-not-control`.

1 Introduction

An oversight system watches an LLM agent's trajectory and decides whether to continue, stop, or intervene. A common approach scores continuation risk, as a failure probability, a confidence, or an uncertainty estimate, and intervenes when that score crosses a threshold. Recent work improves these scores' calibration [Kadavath et al., 2022, Xuan et al., 2026, Zhang et al., 2026c], and routing interventions on them is common practice [Ren et al., 2023, Ding et al., 2026]. Consider two states assigned the same failure score, with a handoff available (Figure 1, middle). In one, the handoff can rescue the episode; in the other, the episode can no longer succeed, whoever acts next. Intervention is worth its cost in the first and worse than quitting in the second. A system that routes on the shared score must either waste the intervention on the second or let the first run to failure.

The score is not wrong. It is answering a different question. Failure probability describes what happens if the agent continues. The intervention decision depends on what changes if it does not: at prefix h, the advantage of intervention i is $\tau_i(h) = Q_i(h) - Q_{\text{continue}}(h)$, the difference in expected downstream utility. This settles the binary choice; with more actions the controller compares all expected utilities. A summary supports the decision only if it preserves that comparison. Recalibration

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

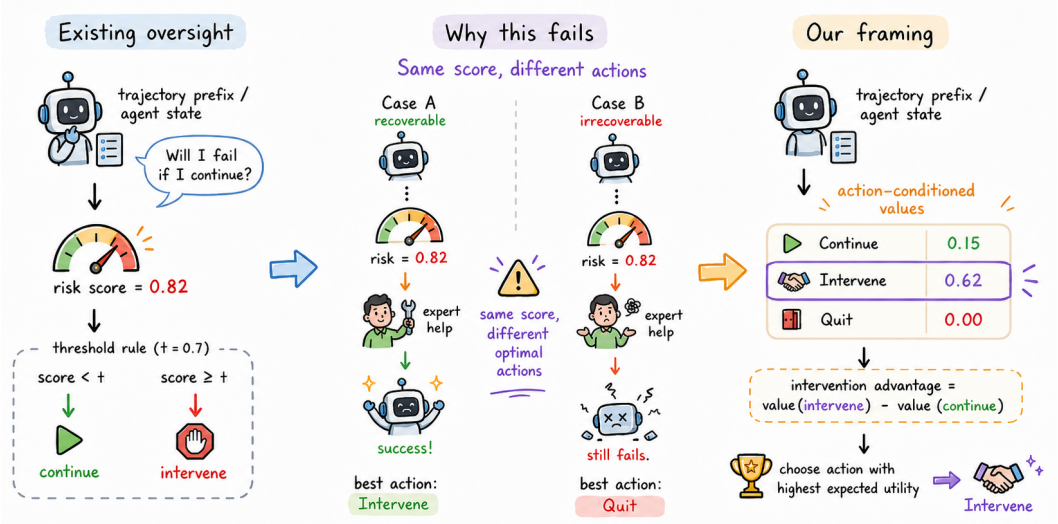

Figure 1: The same risk can call for different actions. Left: failure-triggered oversight scores continuation risk and intervenes when the score crosses a threshold t. Middle: Cases A and B have the same risk score, but a handoff can rescue only A, so the best action is to intervene in A and to quit in B. Right: for a generic state, the decision compares the expected utility of each available action; the intervention advantage is the difference between intervening and continuing. Values are illustrative, with quitting worth zero as in our utilities.

cannot restore a distinction the score lacks: a strictly increasing transform leaves the score's order, and the threshold policies it can express, unchanged (Remark 1).

Two lines of work come close. Learning to defer puts the expert's accuracy into the deferral objective [Mozannar and Sontag, 2020], and Vasudev et al. [2026] show for LLM agents that accurate failure prediction can coexist with harmful intervention. Neither measures how much of a risk score's control loss is recovered at the same state by changing what the monitor is trained on, or whether a controller trained on intervention outcomes keeps its lead online. A clean answer requires three things: outcomes for the actions not taken, at the same state; a pipeline in which only the supervision target changes; and a deployment with a real intervention, where the controller's own decisions shape the trajectory.

We meet the first requirement by prefix branching. In a replayable environment we reconstruct a decision prefix, execute continuation and every available alternative from it, and record their downstream utilities (Section 3). These outcomes supervise controllers that see only the prefix; the deployed controller never runs a trial intervention. The matched comparison of Section 5.2 meets the second requirement, and the closed-loop suite of Section 5.4 the third. Section 5 establishes five findings.

Failure-triggered routing leaves most utility unclaimed. Across 26 ALFWorld runs, its regret is 0.506 against 0.110 for a prefix-feature controller, which reads the prefix features directly, and failure-score thresholds and action order chosen on the test outcomes still trail a single intervention-trained scalar (0.275 vs. 0.229; Section 5.1).

The supervision target accounts for the loss. Holding features, estimator, and router fixed, changing only the label from failure to intervention utility moves regret from 0.513 to 0.094, and the gain replicates on a second suite of mid-episode prefixes (Section 5.2).

Calibration cannot recover it. Above a failure score of 0.7, the best action is to intervene on 46% of prefixes and to quit on 48%, interleaved along the score; Platt scaling cuts the confidence score's calibration error from 0.463 to 0.006 and leaves its control regret unchanged (Section 5.3).

The gain survives a real handoff. On 300 unseen tasks with GPT-5.4 as the handoff, the prefix-feature controller hands off on 35% rather than 48% of episodes and succeeds on 45% rather than 37% (Section 5.4).

The gap depends on the regime, and its supervision is affordable. It is largest on ALFWorld, where the expert can rescue many high-risk prefixes, smaller on ScienceWorld, and small on two reasoning

benchmarks, where a gold-answer probe separates a weak intervention from a score that already decides (Section 5.5); a quarter of the intervention branches retains 92% of the gain (Section 5.6).

Section 2 formalizes when a summary is sufficient for control and what is lost when it is not. This criterion, the branching protocol, and the measurements above are our contributions.

2 When Is a Summary Sufficient for Control?

This section makes the introduction's claim precise: what a summary must preserve for the decision to be lossless, and how much is lost when it does not.

2.1 Prediction targets and control targets

At each prefix state $h \in \mathcal{H}$ of an agent trajectory, a controller chooses from $\mathcal{A} = \{\text{continue}, a_1, \ldots, a_K\}$, where the a_k are non-continue actions such as handoff, verification, repair, or quitting. Let $U_a(h)$ be the random downstream utility after choosing a at h, costs included, and $Q_a(h) = \mathbb{E}[U_a(h) \mid h]$. Two quantities are central:

$$p_{\text{fail}}(h) = \Pr\big(U_{\text{continue}}(h) \le 0 \mid h\big), \qquad \tau_i(h) = Q_i(h) - Q_{\text{continue}}(h). \tag{1}$$

The first asks whether continuation is likely to fail. The second asks whether intervention i improves expected utility over continuing. We call $Q_i(h)$ the intervention value at h and $\tau_i(h)$ its advantage over continuing; $p_{\text{fail}}(h)$ describes the continuation branch alone. The optimal action is $A^*(h) = \arg\max_{a \in \mathcal{A}} Q_a(h)$; for the binary set $\{\text{continue}, \text{intervene}\}$ it is determined by the sign of $\tau(h)$, and with quitting or several interventions available it selects among all $Q_a(h)$ (the main experiments use three actions).

A summary $g(h)$ carries enough for the decision when conditioning on it never forces the controller to merge states whose optimal actions differ; we take this as the definition.

Definition 1 (Sufficiency)**.** A measurable summary $g : \mathcal{H} \to \mathbb{R}^m$ is sufficient for control if there exists a measurable d with $\Pr_{h \sim \mathcal{D}}[d(g(h)) \in A^*(h)] = 1$.

Failure-triggered routing is the special case $d(p_{\text{fail}}(h))$, and routing on a confidence or an uncertainty score is the same special case with a different g. In Figure 1, Cases A and B share a risk score of 0.82 while their optimal actions differ, so no rule applied to that score is right at both.

Proposition 1 (Lossless routing, binary case)**.** *Under $\mathcal{A} = \{\text{continue}, \text{intervene}\}$, g is sufficient if and only if the sign of $\tau(h)$ is recoverable from $g(h)$ up to the tie set $\{\tau = 0\}$: there is a g-measurable region E with $\{\tau > 0\} \subseteq E \subseteq \{\tau \ge 0\}$ up to $\mathcal{D}$-null sets.*

A summary need not represent the trajectory; it must preserve the decision boundary induced by intervention advantage (proof in Appendix A). A failure score can correlate with τ and still fail this test, because τ depends on how the intervention interacts with the state, which p_{fail} does not describe.

2.2 The value lost when sufficiency fails

Proposition 1 is all-or-nothing: it says whether a summary can be lossless, not how much is lost when it cannot. Let $V^* = \mathbb{E}[\max_a Q_a(h)]$ be the value of the fully informed controller and $V_g = \mathbb{E}[\max_a \mathbb{E}[Q_a(h) \mid g(h)]]$ the value of the best controller measurable with respect to g. The *abstraction loss* is

$$\text{Gap}(g) = V^* - V_g \ge 0. \tag{2}$$

It is zero if and only if g is sufficient. It is not an estimation error: a controller with perfect conditional expectations incurs it because it must average over states that g cannot tell apart, such as Cases A and B of Figure 1 under the failure score. Appendix A lower-bounds it by the mass with different optimal actions that shares a value of g (Proposition 3).

Recalibration is the natural response to a lossy score; Remark 1 says what it can and cannot change.

Remark 1 (Recalibration preserves the threshold policy class). For a fixed score g and a strictly increasing continuous recalibration T, $\{h : T(g(h)) > T(t)\} = \{h : g(h) > t\}$. With thresholds chosen freely, the threshold policies expressible on $T \circ g$ are the same as on g, and so is the best value within that class; the same holds for several thresholds with a fixed action ordering. It concerns

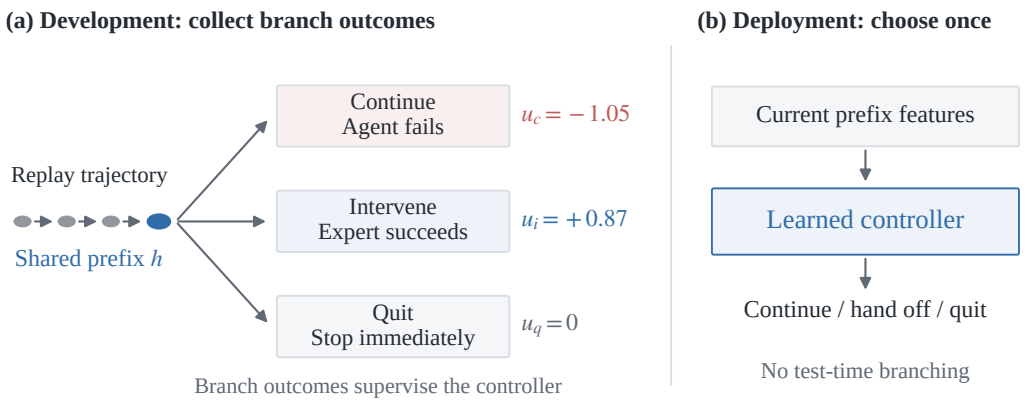

Figure 2: Collect alternatives offline; choose one action online. (a) Replay to a shared prefix and execute each branch to obtain supervision; the utilities are illustrative and correspond to Case A of Figure 1. (b) The deployed controller sees prefix features and selects one action, without executing trial branches.

representable policies, not a fixed numerical threshold, a finite-sample threshold search, or rules that use probabilities and costs directly (Appendix A).

How much $\mathrm{Gap}(g)$ costs in practice depends on two conditions. Some alternative must have positive advantage on a non-null set, since otherwise no controller gains from deviating from continuation, and g must merge states with different optimal actions, since otherwise some g-measurable routing is lossless (threshold routing also needs the action regions to be ordered in g). Section 5 examines both on branched data.

3 Prefix Branching

Measuring what a summary loses requires knowing, at the same state, what each action would have yielded. Ordinary logs reveal only the outcome of the action taken, so they cannot identify $\tau(h)$ without assumptions that would themselves become suspects in the comparison. Prefix branching removes that dependence by supplying the missing outcomes directly, in environments that can be replayed (Figure 2). We collect base trajectories, select decision prefixes, and replay the agent to each prefix, keeping a replay only if it agrees with the original run step by step (retained interactive data match at 100%). From each verified prefix we execute every candidate action, so the same prefix yields outcomes for continue, the deployed intervention, and quit, which define the oracle action and the regret used throughout. Figure 2 traces Case A of Figure 1 through this protocol: continuation fails, the handoff succeeds, and quitting yields zero, so the oracle action is to intervene; in Case B the handoff also fails, and the oracle action is to quit.

Branching is a development-time protocol; the deployed controller does not branch. Prefixes are split into training, validation, and test sets, with all prefixes from one base trajectory assigned to the same split. In the ALFWorld suite no test task appears in any run's training or validation split, within or across models and seeds; the 26 runs cover 84 distinct held-out tasks, each used by 3.4 runs on average (Appendix C.3).

Each branch is a single execution, so a realized branch utility is one sample of $U_a(h)$; repeating both actions five times on 72 ALFWorld and 72 ScienceWorld test prefixes changes the oracle action on 4.2% and 8.3% of prefixes, mostly near ties, and the main regret reductions by 0.001 (Appendix C.7).

4 Controllers and Offline Evaluation

4.1 From branch outcomes to deployed decisions

Branching gives each development prefix h a realized utility $u_a(h)$ per evaluated action, with the intervention and the execution policy after each action fixed within a suite. Predictors are fit on training prefixes, configurations selected on validation utility, and test branch outcomes reserved for evaluation.

Because the comparison is between summaries, we keep the controller family simple: random-forest predictors combined by one of four decision rules (direct value, explicit expected utility, expected utility with a lower confidence bound, and a two-stage rule), with the rule selected per run on validation (Appendix C.5); a stronger architecture would blur whether a gain came from the summary or from the model that reads it. Within that family we distinguish a controller's prediction target, its deployed input, and its action-selection rule. A continuation-risk predictor estimates $p_{\text{fail}}(h) = \Pr(U_{\text{continue}} \leq 0 \mid h)$; we write $\hat{p}_{\text{fail}}(h)$ for its learned output. An intervention-aligned predictor estimates an outcome or utility quantity involving an available intervention. Intervention-success probability, intervention utility, continuation utility, and their difference are distinct targets; p_{fail} is a function of the continuation branch alone, so it carries one term of τ and not the other. A compact summary combines a few predicted quantities; a prefix-feature controller uses the prefix feature vector directly. All summaries and the prefix-feature controller use prefix-available features and receive no intervention preview; targets and feature lists are given in Appendix C.5.

Benchmarks, agents, and interventions. ALFWorld [Shridhar et al., 2021] and ScienceWorld [Wang et al., 2022a] are interactive environments where an intervention changes the environment state; GSM8K [Cobbe et al., 2021] and HotpotQA [Yang et al., 2018] are reasoning tasks where it changes only the reasoning branch. Deployed interventions differ in strength by design, so that the two conditions of Section 2.2 can fail separately rather than together: defer-to-expert (ALFWorld), gold-prefix handoff (ScienceWorld), and same-model verify-and-reanswer (GSM8K, HotpotQA), defined in Appendix C.2. Base agents are Qwen2.5-7B-Instruct [Qwen Team, 2025] and Llama-3.1-8B-Instruct [Grattafiori et al., 2024]. Utility is outcome score minus intervention and per-step costs, with quit worth zero (Table 4); ScienceWorld's continuous score gives the failure label $U_{\text{continue}} \leq 0$, with a boundary sweep in Appendix C.9.

4.2 Comparison protocol and offline regret

Every deployable pipeline is trained on the same prefixes, tuned on the same validation split, and evaluated on the same held-out branches, so differences between pipelines are paired. Validation splits are small (4 to 20 prefixes per run), so selection is noisy; the matched comparison of Section 5.2 therefore fixes the pipeline and gives every target the same two-threshold search (Appendix D.1). For a frozen policy π, the offline evaluation measures regret relative to the best realized branch:

$$\widehat{R}(\pi) = \frac{1}{N} \sum_{j=1}^{N} \left[\max_{a \in \mathcal{A}} u_a(h_j) - u_{\pi(h_j)}(h_j) \right]. \tag{3}$$

Regret is computed within each run and aggregated per suite (Appendix C.6). Since $\mathbb{E}[\max_a U_a \mid h] \geq \max_a \mathbb{E}[U_a \mid h]$, the realized oracle inflates absolute regret relative to Bayes regret for policies independent of the test outcomes; the common term cancels in differences between policies evaluated on the same branches, so we report paired differences with their confidence intervals; Appendix C.11 lists the protocol behind each table.

5 Results

The offline comparisons of Sections 5.1 to 5.3 use an oracle expert as the intervention, so that intervention quality does not limit what a summary can recover; Section 5.4 replaces it with a real handoff.

5.1 Failure-triggered routing leaves most utility unclaimed

On the main ALFWorld suite, failure-triggered routing has regret 0.506 and the prefix-feature controller 0.110 (Table 1). The reduction is 0.396, with a 95% interval of $[0.310, 0.488]$ over runs; weighting every held-out prefix equally, it is 0.358, with a task-clustered interval of $[0.315, 0.404]$ over the 84 held-out tasks. The failure score predicts outcomes well (AUROC 0.74, calibration error 0.06), and better thresholds do not close the gap: a random-forest controller on the same score lowers regret only to 0.449, and thresholds and action order chosen on the test outcomes reach 0.275, still above a single intervention-trained scalar selected on validation (0.229). A compact summary of a few predicted quantities reaches 0.161, a reduction of 0.345 $[0.279, 0.420]$, and prefix-feature control improves on it by a further 0.051 $[0.028, 0.078]$.

Table 1: Comparison ladder on the main ALFWorld suite. The 26 runs share evaluation prefixes and an oracle-expert intervention. Deployable rows select on validation; the shaded reference selects thresholds and action order on test outcomes. Inputs and controllers vary here; Table 2 holds the scalar pipeline fixed.

Deployed input	Controller / selection	Regret ↓
Failure score	Two-threshold rule	0.506
Failure score	Random forest with lower-confidence adjustment	0.449
Failure score	*Two thresholds + action order, selected on test*	*0.275*
Intervention-aligned scalar	Prefix-only predictor and routing rule	0.229
Compact summary	Predictor combination and routing rule	0.161
Prefix feature vector	Action-conditioned controller family	**0.110**

5.2 The supervision target accounts for the loss

The pipelines in Table 1 differ in input, controller, and selection at once, so the gap could come from any of them. Table 2 holds the scalar pipeline fixed, with the same 26 features, random-forest regressor, and validation-selected two-threshold router, and changes only the supervision label.

Table 2: Changing the supervision target under a fixed pipeline. All scalar rows use the same 26 features, random-forest regressor, and validation-selected two-threshold router; the last row uses a pair router. Means are over runs; Δ is the reduction from the failure label, with paired-bootstrap 95% intervals. Protocols: Appendix D.1 and D.2.

	Main suite, 26 runs		Mid-episode, 8 runs	
Supervision target	Regret ↓	Δ [95% CI]	Regret ↓	Δ [95% CI]
Failure label	0.513	—	0.295	—
Continuation utility	0.451	0.062 $[-0.051, 0.175]$	0.245	0.049 $[-0.005, 0.110]$
Intervention success	0.124	0.389 $[0.297, 0.472]$	**0.081**	0.214 $[0.178, 0.257]$
Advantage $U_i - U_{\text{continue}}$	0.152	0.361 $[0.289, 0.434]$	0.148	0.147 $[0.077, 0.222]$
Intervention utility U_i	0.094	0.419 $[0.340, 0.501]$	0.098	0.196 $[0.165, 0.241]$
Both values (pair router)	**0.069**	0.445 $[0.397, 0.493]$	0.095	0.199 $[0.152, 0.255]$

On the main suite, continuation-side targets leave regret at 0.513 (failure) and 0.451 (utility). Every intervention-side scalar removes at least 0.36: intervention success reaches 0.124, advantage 0.152, and intervention utility 0.094. Improvements occur in 24 to 25 of 26 runs, with intervals over runs and over the 84 held-out tasks excluding zero (classifier variant in Appendix D.1). With the pipeline fixed, the target accounts for the gap.

A single difference cannot express quitting, which is optimal only when both values are non-positive. Here the expert's value tracks recoverability, so low intervention value also signals a failing continuation. This is consistent with intervention utility beating advantage and both values doing best. Every intervention-side target keeps its edge across noisy experts; with the expert truncated to two or three steps, only the two-value pair still improves on the failure label (Appendix D.1).

The right half of Table 2 repeats the ablation at six evenly spaced prefixes per trajectory, mostly mid-episode, with 57 to 60 validation prefixes per run (Appendix D.2). The gain from intervention-side supervision replicates: failure-triggered routing has regret 0.286, intervening at every prefix 0.263, and the validation-selected controller 0.088, improving all eight runs.

5.3 Calibration cannot separate what the score merges

Calibrating the failure score does not change which prefixes it groups together. Figure 3 plots the branched outcomes behind Table 1. Above a learned failure score of 0.7, prefixes where intervention pays and prefixes where it fails are interleaved along the score axis: the best action is to intervene on 46% of them and to quit on 48%. Cases A and B of Figure 1 sketch one such pair. A strictly

increasing recalibration only relabels the horizontal axis. It moves the points without reordering them, so the interleaving, and the threshold policies the score can express, stay the same (Remark 1).

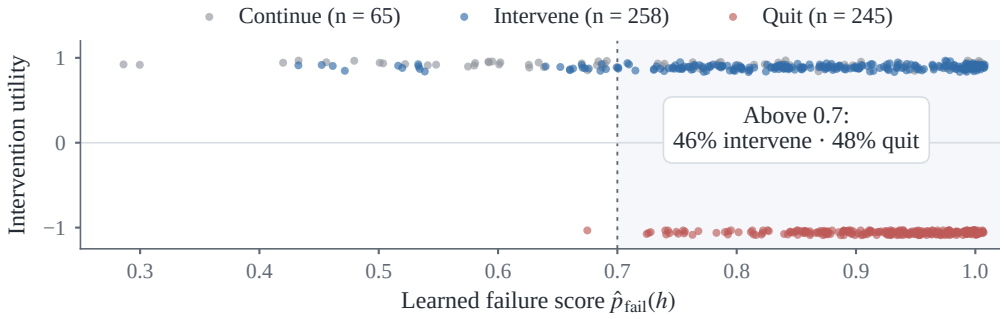

Figure 3: High failure risk hides opposite intervention outcomes. ALFWorld main-suite prefixes, colored by the realized-best action and jittered for visibility. Above a learned failure score of 0.7, the best action is intervene on 46% of prefixes and quit on 48%. A strictly increasing recalibration rescales the horizontal axis without reordering the points.

On the main-suite ALFWorld rows pooled across runs (Table 10), Platt scaling cuts the confidence score's calibration error from 0.463 to 0.006 and leaves its control regret at 0.318 after thresholds are reselected; isotonic regression, which creates ties, raises the failure score's regret from 0.358 to 0.462. Training on intervention outcomes changes the order itself; Table 2 measures what that recovers.

5.4 Fewer handoffs, more successes online

The offline suites use an oracle expert and fixed branch outcomes. Does the gain survive a practical intervention, when the controller's own decisions shape the trajectory?

Setup. The closed-loop suite uses 300 fresh ALFWorld task instances that do not overlap training or validation tasks, with 50 instances in each of six model-seed runs. Controllers are trained on offline data collected with the same GPT-5.4 [OpenAI, 2026] handoff intervention and frozen before testing; within each run, policies share task instances and environment seeds. After each base-agent step, the controller chooses continuation, handoff, or quitting; the offline continuation branch instead lets the base agent run to the end of the episode, so branch values are those of committing to an action at h. Once handoff occurs, the stronger model completes the episode.

Net utility is the primary control objective. For an episode with T environment steps,

$$U = r_{\text{terminal}} - 0.05\,\mathbf{1}\{\text{handoff}\} - 0.01\,T, \tag{4}$$

where r_{terminal} is 1 for success, -1 for terminal failure, and 0 for quitting. The symmetric terminal reward is deliberate: a policy cannot raise utility by handing off indiscriminately (the always-hand-off policy in Figure 4). Closed-loop confidence intervals use a paired bootstrap of the six run-level differences with 10,000 resamples.

Results. Handing off more often does not buy more success here. Failure-triggered routing hands off on 48% of episodes and succeeds on 37%. The prefix-feature controller hands off on 35% and succeeds on 45%. Handing off on every episode succeeds on 40%, less often than handing off on a third of them.

Net utility, the control objective, orders the three learned routing pipelines the same way. The prefix-feature controller improves mean utility over failure-triggered routing by 0.166, with a paired-bootstrap 95% interval of $[0.153, 0.176]$; the compact summary improves it by 0.089 $[0.073, 0.108]$; the prefix-feature controller improves over the compact summary by 0.077 $[0.060, 0.094]$. All three improvements are positive in every run. The utility scale is compressed because a terminal failure costs what a success earns, so the success and handoff rates in Figure 4 carry the comparison that utility summarizes.

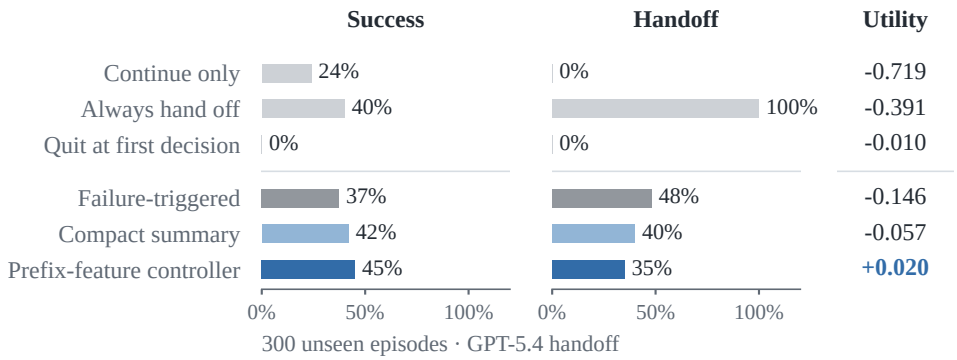

Figure 4: Higher success with fewer handoffs. Closed-loop ALFWorld with a fixed GPT-5.4 handoff, pooled over 300 unseen episodes in six runs. Utility follows Eq. (4); quitting at the first decision is an analytical reference. Full results: Table 5.

5.5 The gap appears where both conditions hold

Section 2.2 predicts a large gap only when the intervention has value and the summary merges states with different optimal actions. The four benchmarks of Table 3 differ in both.

Table 3: Control gains across four benchmarks. FT: failure-triggered routing; Full: prefix-feature controller. Δ is the regret reduction, with paired-bootstrap 95% intervals over runs; bold reductions exclude zero. Absolute regret is comparable only within a benchmark.

Benchmark	Intervention	FT	Full	Δ [95% CI]
ALFWorld	Expert handoff	0.506	0.110	**0.396** $[0.310, 0.488]$
ScienceWorld	Gold-prefix handoff	0.245	0.169	**0.076** $[0.030, 0.130]$
GSM8K	Same-model verify	0.423	0.394	**0.029** $[0.006, 0.053]$
HotpotQA	Same-model verify	0.436	0.417	0.019 $[-0.020, 0.061]$

ALFWorld is consistent with both. The expert intervention has strongly positive realized utility on many high-risk prefixes (Figure 3), and the gain is positive in all 25 cost cells over the 56 ALFWorld runs, including those with degraded experts (Table 13), and survives a non-oracle GPT-5.4 handoff in the closed-loop suite (Section 5.4).[1] ScienceWorld shows a smaller but reliable gain under a strong intervention, positive in 23 of 25 cost cells.

The two reasoning benchmarks show small deployed gains. A small gain admits two readings: the score may already separate the actions, or the intervention may be too weak for any router to matter. We probe this by replacing the same-model verify branch with a gold-answer upper bound and recomputing the controllers on the counterfactual data, with the prefixes and the failure score unchanged (Appendix D.7). With a fixed controller, GSM8K's gain rises from 0.028 to 0.127, while HotpotQA's moves only from 0.043 to 0.052 (Table 12). Both deployed gains are small; under the probe they diverge. This is consistent with a weak intervention limiting GSM8K and with the failure score carrying more of the decision on HotpotQA. At the weak end, prompt-only same-model repair completes the task on up to three of eight branches in an eight-prefix ALFWorld pilot but beats continuation on at most one, and every validation-selected controller chose quit (Appendix D.6). Appendix D.9 gives a branched-data diagnostic for where the gap appears; Appendix D.15 reports cross-model transfer.

5.6 A quarter of the branches retains 92% of the gain

Every result so far assumes that each prefix was branched on every action, and branching is the price of the supervision. In our profiling, collecting both suffixes costs 2.08 times a continuation-only

[1] An earlier version of this paper reported a cross-model repair condition; we removed it because its failure-triggered baseline did not follow the deployed pipeline.

suffix on ALFWorld and 2.04 times on ScienceWorld. That is a development-time cost; the deployed controller decides in 1.4 ms on a CPU. We measure how much branching the controller needs in a separate ALFWorld suite with four actions: continue, same-model replan, GPT-5.4 handoff, and quit (Figure 5). A quarter of the additional branches, chosen by uncertainty, retains 92% of the full-data gain; the remaining 0.020 has a 95% interval of $[0.010, 0.030]$.

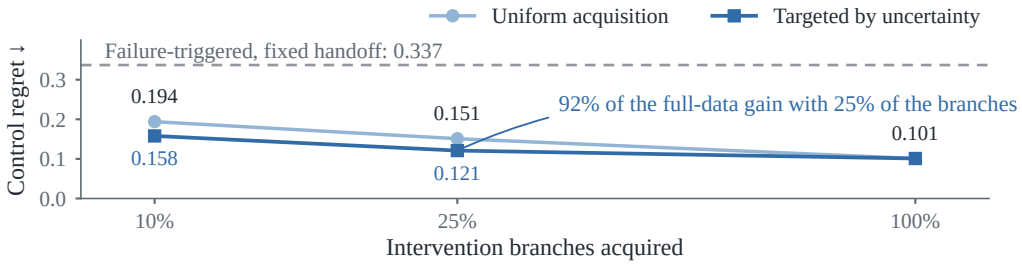

Figure 5: A quarter of the branches retains 92% of the gain. Four-action ALFWorld suite, 12 runs. Targeted acquisition selects prefix-action pairs by uncertainty. This suite has a different action set and oracle from the main suite; full results appear in Table 6.

6 Related Work

Routing on continuation-risk scores. Confidence and uncertainty estimation provide signals for oversight [Kadavath et al., 2022, Farquhar et al., 2024, Xuan et al., 2026, Zhang et al., 2026c]. Ren et al. [2023] use calibrated uncertainty to decide when to seek help, and Ding et al. [2026] use calibrated beliefs for cost-aware exploration. Closest in setting, Wang et al. [2026] train a prefix-based failure monitor for SWE agents and pair it with a same-policy restart, showing that an alarm's value depends on the recovery action. Pinto [2026] examines how a trusted monitor's deferral accuracy depends on the policy lineage it was calibrated on. Related studies examine failures to turn evidence judgments into stopping decisions [Zhang et al., 2026a] and the dependence of tool-use gains on feedback reliability [Zhang et al., 2026b]. Our comparison isolates the supervision target with features, estimator, and router fixed, asking whether the resulting signal retains the action comparison that intervention requires.

Deferral and intervention value. Learning to defer studies how a predictor should allocate decisions to a downstream expert, with expert performance entering the objective [Mozannar and Sontag, 2020, Madras et al., 2018, Mao et al., 2023, Verma et al., 2023]. This is the closest formal precedent for our comparison; a predicted intervention-success scalar, one of the candidate summaries in Appendix D.3, is a deferral-style estimate of expert accuracy. In an agent trajectory that outcome is not observed unless the action is executed, which is why we branch. For fixed intervention and continuation policies, $Q_i(h) - Q_{\text{continue}}(h)$ is a conditional effect on downstream utility, connecting the setup to uplift modeling [Gutierrez and Gérardy, 2017]; Tsiourvas et al. [2025] learn query-level LLM routing by minimizing regret from observational data, whereas our decisions are taken at trajectory prefixes with executed alternatives. Vasudev et al. [2026] establish a prediction-prevention gap for LLM agents. What neither line supplies is the measurement in Table 2: with the pipeline fixed, changing the supervision target from failure to intervention utility removes most of the control loss, and a prefix-feature controller trained on those outcomes keeps its lead over failure-triggered routing online.

Richer signals. Selective prediction [Geifman and El-Yaniv, 2017, 2019], reasoning-time branch selection [Wang et al., 2022b, Yao et al., 2023, Snell et al., 2025], process-level verifiers [Lightman et al., 2024, Uesato et al., 2022], post-hoc failure attribution [Yeh et al., 2026], and test-time orchestration [Qin et al., 2026] supply richer signals than a single risk score; our criterion applies to each, since a signal supports an intervention decision to the extent that it preserves the comparison between actions. Complementary work studies typed skill selection [Bai et al., 2026], programmatic skill induction [Shao et al., 2026], and memory-guided counterfactual planning [Chi et al., 2026]. Recent agent work also studies runtime monitors and guards [Panthi and Abdelfattah, 2026, Zheng et al., 2026, Fayyazi et al., 2026, Liu et al., 2026], recovery and intervention [Zhao et al., 2026, Ji

et al., 2026, Yu et al., 2026, Chen et al., 2026, Zhang et al., 2026d], model routing and escalation [Yan et al., 2026, Lei and Huang, 2026, Ghasemabadi et al., 2026], and failure localization and action credit [Li et al., 2026, Yang and Huang, 2026].

7 Conclusion

Oversight is a decision, and a failure score answers only half of it. A monitor trained to predict failure tells the controller when continuing is bad, not whether anything better is available. On ALFWorld, changing what the monitor is trained on, from failure to intervention outcomes, accounts for most of the control loss with the pipeline fixed, on two prefix distributions. Recalibrating the failure score cannot do this, because it does not reorder states. A single intervention-side scalar removes most regret with the expert and its noisy variants; under truncation only the two-value predictor still improves on the failure label. The prefix-feature controller's lead over failure-triggered routing survives online with a real stronger-model handoff. Oversight signals should be evaluated by task utility, success, and cost alongside predictive quality.

Limitations and future work. The in-sample reference of Table 1 bounds two-threshold rules on the learned score, not every rule on it. Online, the prefix-feature controller's utility exceeds the quit reference by 0.030 without an interval, and without a learned continue-or-quit controller or a fixed handoff budget the online gain is not attributed between better stopping and better handoffs. The gold-answer probe changes the intervention and the supervision, and with an intervention that always succeeds the failure probability is itself sufficient, so its added gain measures estimation and routing rather than missing information (Appendix D.7). The budget result is confined to the four-action suite, whose failure-triggered row cannot choose replan. Strong results are concentrated on ALFWorld. The next experiments follow from these limits: a matched-target ablation with a real handoff, a learned continue-or-quit controller under a fixed handoff budget, and agents whose recovery actions differ from a handoff, such as restarts for software agents. As with other oversight methods, more reliable intervention decisions could also make a misused agent more reliable.

Acknowledgments and Disclosure of Funding

This research/project is supported by the National Research Foundation, Singapore under its National Large Language Models Funding Initiative (AISG Award No: AISG-NMLP-2024-003). Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not reflect the views of National Research Foundation, Singapore. The authors declare no competing interests.

References

Tong Bai, Zhenglin Wan, Pengfei Zhou, Xingrui Yu, Yang You, and Ivor W. Tsang. SkillDAG: Self-evolving typed skill graphs for LLM skill selection at scale. *arXiv preprint arXiv:2606.03056*, 2026. URL `https://arxiv.org/abs/2606.03056`.

Chaoran Chen, Vy Nguyen, Ziji Zhang, Abhinav Gullapalli, Ziyi Wang, Yuxuan Lu, Dakuo Wang, Jing Huang, Zhou Yu, and Jin Lai. Bench2Robust: Scenario-controlled recovery for tool-using agents. *arXiv preprint arXiv:2608.11977*, 2026. URL `https://arxiv.org/abs/2608.11977`.

Haotian Chi, Zeyu Feng, Xingrui Yu, Linbo Luo, Yew-Soon Ong, Ivor Tsang, Hechang Chen, Yi Chang, and Haiyan Yin. EvoCF: Multi-agent collaboration via agentic memory-driven evolutionary counterfactual planning. In *Proceedings of the 43rd International Conference on Machine Learning*, volume 306 of *Proceedings of Machine Learning Research*, pages 19424–19437. PMLR, 2026. URL `https://proceedings.mlr.press/v306/chi26b.html`.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Wenxuan Ding, Nicholas Tomlin, and Greg Durrett. Calibrate-then-act: Cost-aware exploration in LLM agents. *arXiv preprint arXiv:2602.16699*, 2026. doi: 10.48550/arXiv.2602.16699. URL https://arxiv.org/abs/2602.16699.

Sebastian Farquhar, Jannik Kossen, Lorenz Kuhn, and Yarin Gal. Detecting hallucinations in large language models using semantic entropy. *Nature*, 2024.

Arya Fayyazi, Mehdi Kamal, and Massoud Pedram. The agent knew: Internal states forecast unauthorized actions and stop them, and the lies that follow. *arXiv preprint arXiv:2610.17301*, 2026. URL https://arxiv.org/abs/2610.17301.

Yonatan Geifman and Ran El-Yaniv. Selective classification for deep neural networks. In *Advances in Neural Information Processing Systems*, 2017.

Yonatan Geifman and Ran El-Yaniv. SelectiveNet: A deep neural network with an integrated reject option. In *International Conference on Machine Learning*, 2019.

Amirhosein Ghasemabadi, Ruichen Chen, Bahador Rashidi, and Di Niu. Multi-head latent control: A unified interface for LLM agent decision making. *arXiv preprint arXiv:2607.14277*, 2026. URL https://arxiv.org/abs/2607.14277.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, et al. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Pierre Gutierrez and Jean-Yves Gérardy. Causal inference and uplift modelling: A review of the literature. In *Proceedings of the 3rd International Conference on Predictive Applications and APIs*, volume 67 of *Proceedings of Machine Learning Research*, pages 1–13. PMLR, 2017. URL https://proceedings.mlr.press/v67/gutierrez17a.html.

Changxiu Ji, Amy Lu, Qizheng Zhang, and Kunle Olukotun. Sentry: Learning to recover from LLM agent failures at test time. *arXiv preprint arXiv:2610.02994*, 2026. URL https://arxiv.org/abs/2610.02994.

Saurav Kadavath, Tom Conerly, Amanda Askell, et al. Language models (mostly) know what they know. *arXiv preprint arXiv:2207.05221*, 2022.

Ji'an Lei and Jian Huang. TACIT-Switch: Cost-aware model escalation for LLM agents from censored supervision. *arXiv preprint arXiv:2608.27911*, 2026. URL https://arxiv.org/abs/2608.27911.

Bingjie Li, Yumeng Song, Zhongming Yao, and Tianyi Li. Localizing emergent failures in agentic AI: Recovering minimal repair families via counterfactual replay. *arXiv preprint arXiv:2608.29228*, 2026. URL https://arxiv.org/abs/2608.29228.

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

Yushan Liu, Peibo Sun, Xintao Chao, Zhenyang Yang, Yifan Xie, Lingfeng Zhang, Shoujie Li, Chenyu Tang, Fang Chen, Xiao-Ping Zhang, and Wenbo Ding. CheckVLA: Execution-time verification with action-conditioned world model for long-horizon mobile manipulation. *arXiv preprint arXiv:2607.26789*, 2026. URL https://arxiv.org/abs/2607.26789.

David Madras, Toni Pitassi, and Richard Zemel. Predict responsibly: Improving fairness and accuracy by learning to defer. In *Advances in Neural Information Processing Systems*, volume 31, 2018. URL https://proceedings.neurips.cc/paper/2018/hash/09d37c08f7b129e96277388757530c72-Abstract.html.

Anqi Mao, Christopher Mohri, Mehryar Mohri, and Yutao Zhong. Two-stage learning to defer with multiple experts. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

Hussein Mozannar and David Sontag. Consistent estimators for learning to defer to an expert. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 7076–7087. PMLR, 2020. URL https://proceedings.mlr.press/v119/mozannar20b.html.

OpenAI. GPT-5.4 Thinking system card, March 2026. URL https://openai.com/index/gpt-5-4-thinking-system-card/.

Sugam Panthi and Rabab Abdelfattah. Outcome monitors: Recovery affordances for silent tool failures. *arXiv preprint arXiv:2608.19303*, 2026. URL https://arxiv.org/abs/2608.19303.

Lucas Pinto. Calibration-family overfit: Why trusted sabotage monitors don't transfer across lineages. *arXiv preprint arXiv:2607.06596*, 2026. URL https://arxiv.org/abs/2607.06596.

Peijia Qin, Qi Cao, and Pengtao Xie. ATLAS: Agentic test-time learning-to-allocate scaling. *arXiv preprint arXiv:2606.01667*, 2026. URL https://arxiv.org/abs/2606.01667.

Qwen Team. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2025.

Allen Z. Ren, Anushri Dixit, Alexandra Bodrova, Sumeet Singh, Stephen Tu, Noah Brown, Peng Xu, Leila Takayama, Fei Xia, Jake Varley, Zhenjia Xu, Dorsa Sadigh, Andy Zeng, and Anirudha Majumdar. Robots that ask for help: Uncertainty alignment for large language model planners. In *Conference on Robot Learning*, 2023.

Jie-Jing Shao, Haiyan Yin, Yueming Lyu, Xingrui Yu, Lan-Zhe Guo, Ivor Tsang, James Kwok, and Yu-Feng Li. Lifting traces to logic: Programmatic skill induction with neuro-symbolic learning for long-horizon agentic tasks. In *Proceedings of the 43rd International Conference on Machine Learning*, volume 306 of *Proceedings of Machine Learning Research*, pages 110314–110337. PMLR, 2026. URL https://proceedings.mlr.press/v306/shao26i.html.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. ALFWorld: Aligning text and embodied environments for interactive learning. In *International Conference on Learning Representations*, 2021.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling LLM test-time compute optimally can be more effective than scaling parameters for reasoning. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/1b623663fd9b874366f3ce019fdfdd44-Abstract-Conference.html.

Asterios Tsiourvas, Wei Sun, and Georgia Perakis. Causal LLM routing: End-to-end regret minimization from observational data. *arXiv preprint arXiv:2505.16037*, 2025. URL https://arxiv.org/abs/2505.16037.

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

Rakshith Vasudev, Melisa Russak, Dan Bikel, and Waseem Alshikh. Accurate failure prediction in agents does not imply effective failure prevention. *arXiv preprint arXiv:2602.03338*, 2026. doi: 10.48550/arXiv.2602.03338. URL https://arxiv.org/abs/2602.03338.

Rajeev Verma, Daniel Barrejon, and Eric Nalisnick. Learning to defer to multiple experts: Consistent surrogate losses, confidence calibration, and conformal ensembles. In *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics*, volume 206 of *Proceedings of Machine Learning Research*, pages 11415–11434. PMLR, 2023. URL https://proceedings.mlr.press/v206/verma23a.html.

Chenyu Wang, Yunbo Lyu, Junda He, Zhou Yang, Chenxing Zhong, Yaniv Harel, and David Lo. Fail-fast, restart-smart: Early failure prediction and restart for SWE agentic tasks. *arXiv preprint arXiv:2608.03222*, 2026. URL https://arxiv.org/abs/2608.03222.

Ruoyao Wang, Peter Jansen, Marc-Alexandre Côté, and Prithviraj Ammanabrolu. ScienceWorld: Is your agent smarter than a 5th grader? In *Conference on Empirical Methods in Natural Language Processing*, 2022a.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*, 2022b.

Weihao Xuan, Qingcheng Zeng, Heli Qi, Yunze Xiao, Junjue Wang, and Naoto Yokoya. The confidence dichotomy: Analyzing and mitigating miscalibration in tool-use agents. *arXiv preprint arXiv:2601.07264*, 2026.

Haotian Yan, Jiaqi Liu, Bingyi Jing, and Tianjun Yao. Response-informed value estimation for cost-aware multi-turn LLM routing. *arXiv preprint arXiv:2610.23239*, 2026. URL https://arxiv.org/abs/2610.23239.

Hongye Yang and Boxiao Huang. When does action credit need updating? *arXiv preprint arXiv:2609.29007*, 2026. URL https://arxiv.org/abs/2609.29007.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018. doi: 10.18653/v1/D18-1259.

Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. WebShop: Towards scalable real-world web interaction with grounded language agents. In *Advances in Neural Information Processing Systems*, volume 35, pages 20744–20757, 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/82ad13ec01f9fe44c01cb91814fd7b8c-Abstract-Conference.html.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *arXiv preprint arXiv:2305.10601*, 2023. URL https://arxiv.org/abs/2305.10601.

Samuel Yeh, Yiwen Zhu, Shaleen Deep, and Sharon Li. Tracing agentic failure from the flow of success. *arXiv preprint arXiv:2607.12747*, 2026. URL https://arxiv.org/abs/2607.12747.

Yi Yu, Liuyi Yao, Yaliang Li, Enshu Wang, and Libing Wu. Rollback the world, keep the reflection: Rollback-induced reflection for long-horizon LLM agents. *arXiv preprint arXiv:2609.18304*, 2026. URL https://arxiv.org/abs/2609.18304.

Chubin Zhang, Zhenglin Wan, Xingrui Yu, Jingxuan Wu, Yaxin Zhou, Ivor Tsang, and Bo An. Judged useless, queried anyway: Tool-using agents rarely turn their own evidence judgments into stopping decisions. *arXiv preprint arXiv:2610.06191*, 2026a. URL https://arxiv.org/abs/2610.06191.

Chubin Zhang, Zhenglin Wan, Xingrui Yu, Pengfei Zhou, Wangbo Zhao, Jingxuan Wu, Yaxin Zhou, and Ivor Tsang. Don't blindly trust it: How unreliable feedback breaks tool-using LLM agents. *arXiv preprint arXiv:2606.21409*, 2026b. URL https://arxiv.org/abs/2606.21409.

Jiaxin Zhang, Caiming Xiong, and Chien-Sheng Wu. Agentic confidence calibration. *arXiv preprint arXiv:2601.15778*, 2026c.

Zheyuan Zhang, Manqing Mao, Hong Wang, Zhuoer Wang, Samson Koelle, Jie Yuan, Yanjun Lin, James Feng, Nikki Lijing Kuang, Yanfang Ye, and Wei Niu. Optimizing what policies learn from: Recoverability-aware rollout intervention learning. *arXiv preprint arXiv:2608.05080*, 2026d. URL https://arxiv.org/abs/2608.05080.

Jiangrui Zhao, Chenglong Li, Meng Zhang, and Xiaoting Du. Learning when and how to intervene: A hindsight-distilled sentinel for coding agents. *arXiv preprint arXiv:2609.39957*, 2026. URL https://arxiv.org/abs/2609.39957.

Zihao Zheng, Jiayu Long, Baichuan Li, and Junyi Yao. Stale does not mean unsafe: Guard precision for tool-using LLM agents under infrastructure state races. *arXiv preprint arXiv:2609.29522*, 2026. URL https://arxiv.org/abs/2609.29522.

A Proofs

A.1 Proof of Proposition 1

Proof. Fix a measurable summary g.

(*If*) Suppose g is sufficient in the sense of Definition 1, so there exists a measurable selector d such that $d(g(h)) \in A^*(h)$ almost surely. Then the g-routed controller $\pi_g(h) = d(g(h))$ selects an optimal action almost surely and is therefore lossless.

(*Only if*) Suppose no such measurable selector exists. Then every g-routed controller $\pi_g(h) = d(g(h))$ fails to lie in $A^*(h)$ on a set of positive probability, so no controller measurable with respect to g can be lossless.

For the binary action set, let $E = \{h : d(g(h)) = \texttt{intervene}\}$ denote the intervention region induced by a g-routed selector. Losslessness requires intervention on every state with $\tau(h) > 0$, allows either action when $\tau(h) = 0$, and forbids intervention on states with $\tau(h) < 0$. Hence any lossless selector must satisfy

$$\{\tau(h) > 0\} \subseteq E \subseteq \{\tau(h) \geq 0\} \qquad \text{almost surely.}$$

Conversely, any g-measurable set E satisfying this inclusion defines a lossless binary selector. If one further restricts to threshold policies on a scalar g, the intervention region must be representable as a threshold set in g, which imposes the ordering condition of Section 2.2. □

A.2 Abstraction loss identity

Proposition 2 (Abstraction loss of routing through a summary)**.** *Let* $V^* = \mathbb{E}[\max_{a \in \mathcal{A}} Q_a(h)]$ *and* $V_g = \mathbb{E}[\max_a \mathbb{E}[Q_a(h) \mid g(h)]]$. *Then*

$$\mathrm{Gap}(g) = V^* - V_g = \mathbb{E}\Big[\mathbb{E}\big[\max_a Q_a(h) \,\big|\, g(h)\big] - \max_a \mathbb{E}\big[Q_a(h) \,\big|\, g(h)\big]\Big] \geq 0.$$

Corollary 1 (Zero loss iff sufficient). $\mathrm{Gap}(g) = 0$ if and only if g is sufficient in the sense of Definition 1.

A.3 Proof of Proposition 2 and Corollary 1

Proof. The optimal unconstrained controller achieves expected utility

$$V^* = \mathbb{E}\big[\max_{a \in \mathcal{A}} Q_a(h)\big] = \mathbb{E}\Big[\mathbb{E}\big[\max_a Q_a(h) \,\big|\, g(h)\big]\Big],$$

where the second equality follows from the tower property of conditional expectation.

The optimal summary-routed controller, at each value of g, must pick the single action with the highest conditional expected utility. Its value is

$$V_g = \mathbb{E}\Big[\max_a \mathbb{E}\big[Q_a(h) \,\big|\, g(h)\big]\Big].$$

The abstraction loss of the best g-routed controller is therefore

$$\mathrm{Gap}(g) = V^* - V_g = \mathbb{E}\Big[\mathbb{E}\big[\max_a Q_a(h) \,\big|\, g(h)\big] - \max_a \mathbb{E}\big[Q_a(h) \,\big|\, g(h)\big]\Big].$$

This proves Proposition 2.

For Corollary 1: if a sufficient selector d exists, then $V_g \geq \mathbb{E}[Q_{d(g(h))}(h)] = \mathbb{E}[\max_a Q_a(h)] = V^*$, and $V_g \leq V^*$ always, so $\mathrm{Gap}(g) = 0$. Conversely, if $\mathrm{Gap}(g) = 0$, the integrand $\mathbb{E}[\max_a Q_a(h) \mid g] - \max_a \mathbb{E}[Q_a(h) \mid g]$ is nonnegative with zero expectation, hence zero almost surely; taking $d(g)$ as a measurable argmax of $\mathbb{E}[Q_a(h) \mid g]$ gives $\mathbb{E}[\max_a Q_a(h) - Q_{d(g(h))}(h) \mid g] = 0$, and since the integrand is nonnegative, $Q_{d(g(h))}(h) = \max_a Q_a(h)$ almost surely, so $d(g(h)) \in A^*(h)$. □

A.4 Conditional conflict lower bound

Proposition 3 (Conditional conflict lower bound)**.** *Let $\mathcal{A}$ be a finite action set, let $Q_a(h) = \mathbb{E}[U_a \mid h]$ be integrable, and fix a measurable summary $g(h)$. Suppose S_a, S_b are disjoint measurable state sets, $a \neq b$, and $\gamma > 0$ such that, almost surely,*

$$Q_a(h) \geq Q_c(h) + \gamma \quad (h \in S_a,\ c \neq a), \qquad Q_b(h) \geq Q_c(h) + \gamma \quad (h \in S_b,\ c \neq b).$$

Write $q_a(g) = \mathbb{E}[\mathbf{1}_{S_a} \mid g]$ and $q_b(g) = \mathbb{E}[\mathbf{1}_{S_b} \mid g]$. Every deterministic policy $\pi(h) = d(g(h))$ satisfies

$$\mathbb{E}\left[\max_{c \in \mathcal{A}} Q_c(h) - Q_{\pi(h)}(h)\right] \geq \gamma\, \mathbb{E}[\min\{q_a(g), q_b(g)\}]\,.$$

Proof. Pointwise regret is at least $\gamma \mathbf{1}_{S_a} \mathbf{1}_{\{d(g) \neq a\}} + \gamma \mathbf{1}_{S_b} \mathbf{1}_{\{d(g) \neq b\}}$. Conditional on g, its expectation is therefore at least $\gamma\big(q_a(g)\mathbf{1}_{\{d(g)\neq a\}} + q_b(g)\mathbf{1}_{\{d(g)\neq b\}}\big)$. If $d(g) = a$, this is $\gamma q_b(g)$; if $d(g) = b$, it is $\gamma q_a(g)$; otherwise it is $\gamma(q_a(g) + q_b(g))$. Each case is at least $\gamma \min\{q_a(g), q_b(g)\}$. Taking expectations proves the result. □

The bound conditions on the observed value of g, not on an interval containing distinct values; a bin-constant policy is covered by taking the observed summary to be the bin label.

A.5 Proof of Remark 1

For strictly increasing T, $T(g(h)) > T(t)$ holds if and only if $g(h) > t$, so every threshold policy on g is a threshold policy on $T \circ g$ and conversely; thresholds outside the range of g give the two constant policies in both cases. With several thresholds and a fixed action ordering, apply the same identity to each threshold. Continuity is a sufficient assumption covering the Platt transform; boundary conventions need separate care for discontinuous monotone maps.

B Simulation validation

To examine the theoretical quantities in a controlled setting where the data-generating process is known, we construct a synthetic environment and compute the binned abstraction loss of the failure score on generated samples. For brevity, we denote this appendix-only quantity by NMG.

B.1 Simulation design

We generate $n = 50{,}000$ prefix states, each characterized by a latent difficulty and recoverability profile. The utility distributions for each action are constructed so that the sufficiency condition of Proposition 1 is violated by design: two states can have the same $p_{\text{fail}}(h)$ but different optimal actions, because one is *hard but recoverable* (intervention succeeds) while the other is *hard and unrecoverable* (intervention also fails).

Core experiment (binary action). The action set is $\{\texttt{continue}, \texttt{intervene}\}$. NMG is the abstraction loss $\mathrm{Gap}(b(p_{\text{fail}}))$ of the binned score on the generated sample, a finite-sample quantity for the binned score rather than the population $\mathrm{Gap}(p_{\text{fail}})$. We compute it by binning states on true p_{fail} (200 bins) and comparing the oracle utility (best action per state) to the best-per-bin failure-triggered utility (best action per bin):

$$\mathrm{NMG}_{\text{binned}} = \frac{1}{n}\sum_{i=1}^{n} \max_a Q_a(h_i) - \frac{1}{n}\sum_{i=1}^{n} Q_{a^*_{\text{bin}(i)}}(h_i) = 0.0383.$$

The action disagreement rate between the oracle and the best failure-triggered rule is 18.1%, indicating that nearly one in five states is misclassified by failure-triggered routing.

We also compute the isotonic regression bound, which fits the best *monotone* mapping from p_{fail} to actions. This is a failure-threshold controller with a monotone f. On this generated sample its binned loss is 0.0395 against 0.0383 for the bin-constant policy; the two policy classes are not nested (a monotone threshold can split a bin, and a bin-constant policy can be non-monotone), so no ordering between them is implied.

Three-action experiment. We extend to $\{\texttt{continue}, \texttt{verify}, \texttt{quit}\}$ with a catastrophic penalty for incorrect continuation ($c = -1.25$). The NMG increases to 0.0670 after changing both the action set and the wrong-answer penalty; the increase is not attributed to the added action alone. The oracle action distribution is: continue 58.1%, verify 20.1%, quit 21.8%.

B.2 Key validation

Learned controller versus binned quantity. We train a failure-triggered controller with a learned failure model (logistic regression on noisy observed features, observe noise $\sigma = 0.35$) and measure its regret. The learned failure-triggered controller achieves regret 0.0388, close to the binned quantity 0.0383. The binned quantity is not a bound on a learner that uses noisy features, since such a score need not be a function of the true-risk bin, and the difference is not attributed to estimation error alone.

Noise sensitivity. In this experiment the latent sample and the bins are held fixed and only the observation noise of the learned score changes, so the binned quantity does not move. We vary the observation noise σ from 0 to 1.0. The binned quantity stays at 0.0383 while the learned regret ranges over 0.037–0.039, and at the two highest noise levels it falls below the binned value (0.0379 and 0.0366), so the binned quantity is not a bound for the learned controller.

C Protocol and Benchmark Details

C.1 Reward structure

In the offline main suite, each benchmark uses a three-action control set with quit worth zero on the remaining suffix; the four-action suite of Section 5.6 and the closed-loop suite, which charges costs as they occur, are defined in their own sections. The utility of action a at prefix state h is:

$$U_a(h) = \begin{cases} (\text{outcome score}) - c_a - s \cdot (\text{branch steps}), & a \in \{\texttt{continue}, \texttt{intervene}\}, \\ 0, & a = \texttt{quit}, \end{cases}$$

where c_a is a fixed intervention cost and s is a per-step cost.

Table 4: Utility parameters per benchmark. Outcome score is $+1$ for correct, $-w$ for incorrect (binary); ScienceWorld uses a continuous environment score $\in [0, 1]$.

Benchmark	Wrong penalty w	Intervention cost c_a	Step cost s	Intervention
ALFWorld	1.0	0.05	0.01	Defer to expert
ScienceWorld	—	0.05	0.01	Defer to gold prefix
GSM8K	0.5	0.10	—	Verify (re-answer)
HotpotQA	0.5	0.10	—	Verify (re-answer)

For GSM8K and HotpotQA, there are no step costs because these are single-turn reasoning tasks. For ScienceWorld, the environment returns a continuous score rather than a binary outcome, so there is no separate wrong penalty; instead, utility equals the environment score minus costs. Quit always yields utility zero. These asymmetries are intentional: intervention carries a positive cost.

C.2 Intervention definitions

ALFWorld. The defer-to-expert intervention hands the episode to the environment's built-in expert planner, which executes its plan from the branched state until the task is solved or the 20-step episode budget is exhausted; the base agent does not resume. An expert failure therefore means that the planner found no plan from that state or could not finish within the remaining steps, for instance after an irreversible action, not that the agent was interrupted. The intervention-quality ladder degrades this expert in two ways: noise variants replace each expert step with a random admissible non-expert command with probability 0.10, 0.25, or 0.40; truncated variants execute two or three expert steps and stop, and the outcome is the episode state at that point. Costs are in Table 4.

ScienceWorld. Decision prefixes are states along the environment's gold action sequence for each task variation, two prefixes per episode under a 40-step limit. Continuation lets the base agent act from that state until the episode ends. The gold-prefix handoff executes the next three gold actions from the branched state and then returns control to the base agent for the rest of the episode; the outcome is the environment score in $[0, 1]$.

GSM8K and HotpotQA. The same-model verify intervention asks the base agent to re-derive its answer in one turn with tool support, a Python expression on GSM8K and a retrieval query on HotpotQA, and returns the re-derived answer.

Closed loop. The handoff target is GPT-5.4 through the OpenAI API; after a handoff the stronger model completes the episode and the base agent does not resume.

C.3 Data splits and pooling

Splits are counted in tasks, with two decision prefixes per task. The interactive runs use $4/2/4$, $6/3/6$, $8/4/8$, or $20/10/20$ tasks (train/validation/test) per model–seed combination, corresponding to $8/4/8$, $12/6/12$, $16/8/16$, or $40/20/40$ prefixes, respectively; the 26 ALFWorld runs have 8 to 40 test prefixes each and 568 in total. For the supporting benchmarks (GSM8K, HotpotQA), the split is $48/24/48$ prefixes, one per question. Two aggregation modes appear in this paper and are named where used: per-run training with run-level aggregation (main suite), and pooled fitting on concatenated rows (marked pooled). In the ALFWorld main suite, a task-identity audit found no test task in the training or validation split of any run, within or across models and seeds. The 84 distinct held-out tasks are each used by 3.4 runs on average. The audit covers this suite only. For the other suites we rely on the trajectory-level split rule of Section 3, and the pooled analyses below concatenate rows across runs without a separate task-level audit. The primary within-model suite pools across seeds and both agent models as follows:

- ALFWorld: 26 runs (7–8B tier), with splits from $4/2/4$ to $20/10/20$ tasks.
- ScienceWorld: 16 runs in the within-model suite, for $n_{\text{test}} = 364$ pooled prefix states.
- GSM8K: 6 runs (3 seeds $\times$ 2 models), $n_{\text{test}} = 288$ pooled prefix states.
- HotpotQA: 6 runs (3 seeds $\times$ 2 models), $n_{\text{test}} = 288$ pooled prefix states.

The exploitability analysis of Section D.9 also includes the 30 ALFWorld runs with degraded experts (Section D.1), giving 56 ALFWorld runs and 84 regimes in total. Multiple scalar and cost sweeps reuse these same branched trajectories; they add evaluation settings, not environment rollouts.

C.4 Agent models

The base agents are Qwen2.5-7B-Instruct [Qwen Team, 2025] and Llama-3.1-8B-Instruct [Grattafiori et al., 2024] on every benchmark. GPT-5.4 [OpenAI, 2026] is used as an intervention, not as a base agent: it is the handoff target in the closed-loop suite (Section 5.4) and in the four-action suite (Section 5.6). All models are accessed via standard inference without any benchmark-specific fine-tuning. The controller (random forest) is trained on trajectory-level features extracted from the agent's interaction history.

C.5 Feature representation

At each prefix state the controller observes a fixed-length feature vector extracted from the agent's trajectory so far. Our *primary* analyses use only *prefix-available* features for both failure-triggered routing and the prefix-feature controller. This keeps the information available to the compared controllers symmetric. Auxiliary preview-aware variants, used only in this appendix, append features of the intervention output. They are reported separately because a preview is information the base comparison does not have.

Feature categories. All benchmarks share several common feature families: (i) *length statistics* of the question/task description, observation, and model response; (ii) *progress indicators* such as step index and history length; (iii) the agent's self-reported *confidence* score (normalized to $[0, 1]$). Benchmark-specific features are summarized below.

- **GSM8K.** *Continue branch:* question and response lengths, count of numbers in the question, answer-format indicators (has final answer tag, decimal, negative), numeric magnitude features (answer value, max/min/mean of question numbers), range-check flags (answer exceeds question range), hedging-word indicator, and 12 math-keyword indicators (`each`, `total`, `percent`, `half`, etc.), plus confidence. *Verify branch (additional):* verify-expression length, tool-execution success flag, verify answer value, continue–verify agreement, answer gap and gap ratio, range-check flags for both answers, answer-in-question flags, and 6 arithmetic-operator indicators ($+, -, \times, /, (,)$).
- **HotpotQA.** *Continue branch:* question and response lengths, answer token count, answer-present flag, answer-in-question overlap, yes/no question and answer indicators, topic-keyword indicators (`film`, `person`, `city`, `year`, etc.), 7 wh-word indicators, plus confidence. *Verify branch (additional):* search-query length, evidence length, retrieval-success flag, top and mean retrieval scores, number of unique retrieved titles, query–question overlap, continue–verify agreement, answer-in-evidence flags, yes/no flags, and answer lengths.
- **ALFWorld.** *Continue branch:* step index, history length, task/observation lengths, number of admissible commands, action word count, 10 action-type indicators (`go to`, `open`, `take`, `put`, `heat`, `cool`, `clean`, `toggle`, `close`, `look`), "nothing happens" flag, 7-dim task-type one-hot, plus confidence. *Verify branch (additional):* step index, admissible-command count, verify-action word count, continue–verify agreement, 8 verify action-type indicators, observation state flags (`closed`, `open`), and 7-dim task-type one-hot.
- **ScienceWorld.** *Continue branch:* step index, task-description length, observation length (words and characters), number of valid actions, action word count, history length, 7 action-type indicators (`go to`, `open`, `close`, `pick up`, `pour`, `use`, `activate`), `look around` flag, "you see" observation flag, plus confidence. *Verify branch (additional):* step index, valid-action count, continue–verify agreement, action-length difference, 2 `go to` flags (continue and verify), `look around` flag, "focus on task" observation flag.

The prefix-only feature dimensionalities are: GSM8K (32), HotpotQA (19), ALFWorld (26), ScienceWorld (17). For the auxiliary preview-aware variants, the intervention branch appends additional features, yielding GSM8K (56 verify), HotpotQA (35), ALFWorld (40), and ScienceWorld (25).

Predictors and routers. Failure and correctness predictors use scikit-learn's `RandomForestClassifier`; utility and advantage predictors use `RandomForestRegressor`. Both use 200 trees, maximum depth 6, and minimum samples per leaf 2. The matched-target ablation also fits binary targets with the regressor to hold the estimator fixed (Section D.1). For the conservative controller, the standard deviation of per-tree predictions supplies the uncertainty estimate in the lower confidence bound. Router search spaces: the two-threshold rule searches both thresholds on a 12-point quantile grid of the validation scores; the value router searches 12 margin quantiles and 13 quit floors in $[-0.2, 0.4]$; the lower-confidence router searches $\beta_{\text{continue}} \in \{0, 0.1, \ldots, 1\}$ and $\beta_i \in \{0, 0.1, \ldots, 1.5\}$. Each is selected by validation utility.

Feature importance. To check that the controller is not relying on opaque feature-engineering artifacts, we refit pooled interactive-benchmark models and inspect grouped random-forest feature importance. Figure 6 should be read as an *auxiliary* preview-analysis figure: failure and continue models rely mostly on trajectory progress and state-shape signals, while preview-aware intervention models place additional weight on intervention-branch features. The figure separates two questions: the main comparisons ask what the prefix alone achieves, and the auxiliary preview-aware analysis asks how much further preview information helps.

C.6 Statistical testing

Unless stated otherwise, significance tests use paired bootstrap with 10,000 resamples at the *run* level; task-clustered sensitivity intervals are identified explicitly. Each resample draws run-level differences with replacement and computes the mean difference. A comparison is marked significant if the 95% confidence interval excludes zero. Differences and intervals are computed before rounding the displayed values. This choice treats a run, rather than an individual prefix, as the statistical unit and respects dependence among prefixes within a run; test tasks recur across runs, so runs are not independent across tasks. As a sensitivity check we resample the 84 held-out ALFWorld

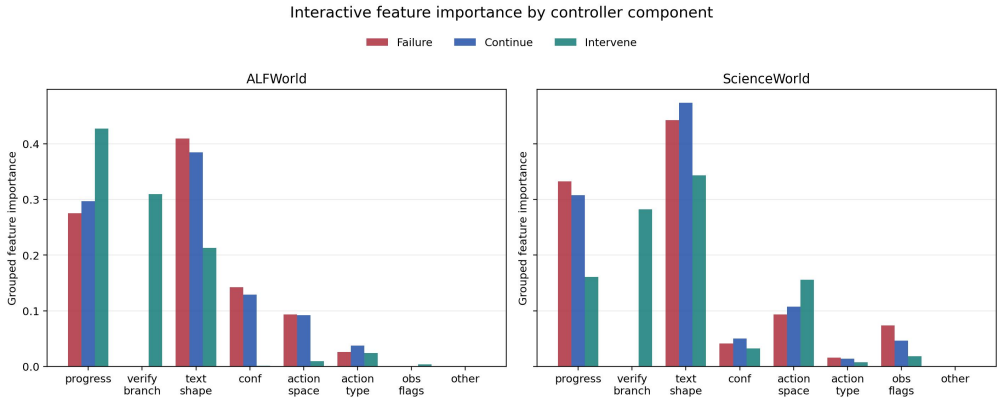

Figure 6: Grouped feature importance for pooled interactive-benchmark models in the auxiliary preview-aware analysis. Failure and continue models depend mostly on trajectory-progress and state-shape signals, whereas preview-aware intervention models also rely on intervention-branch features.

tasks instead of the 26 runs, keeping all rows of a task together: the main regret reduction has a task-clustered 95% interval of $[0.315, 0.404]$ around a row-weighted estimate of 0.358, against the run-level $[0.310, 0.488]$ around the run-weighted 0.396 of Table 1; the two estimates differ because the runs with 8 to 16 test prefixes show larger reductions than the runs with 40. Task-clustered intervals for the matched-target differences are given in Section D.1.

C.7 Repeated branching

Each branch in the main experiments is executed once. To measure within-prefix sampling noise we collected 72 ALFWorld and 72 ScienceWorld test prefixes, balanced across two base models and four seeds, and independently reran both `continue` and `intervene` five times from every prefix, keeping both learned controllers fixed and recomputing branch utilities, oracle actions, and regret from the five-run means. Oracle-action labels agree between the single-rollout and five-run estimates on 69/72 ALFWorld and 66/72 ScienceWorld prefixes, and the sign of intervention advantage on 70/72 and 68/72; disagreements concentrate near ties, and on prefixes with five-run $|\tau(h)| > 0.10$ agreement is 69/70 and 66/68. On these 72 prefixes, the regret reduction from failure-triggered routing to the prefix-feature controller changes from 0.396 (0.508 vs 0.112) to 0.395 (0.501 vs 0.106) on ALFWorld and from 0.075 (0.246 vs 0.171) to 0.074 (0.240 vs 0.166) on ScienceWorld.

C.8 Collection and deployment cost

Profiled on 100 prefixes per interactive benchmark, collecting both suffixes takes 2.08 times the wall-clock time of a continuation-only suffix on ALFWorld and 2.04 times on ScienceWorld. The deployed controller evaluates learned action values in 1.4 ms per decision on a CPU. Open-weight 7–8B inference was run with local vLLM workers on a server with four NVIDIA RTX A5000 GPUs (24 GB each), an AMD EPYC 7443P CPU (24 cores, 48 hardware threads), and 188 GiB RAM. Each vLLM worker used a single GPU; controller training used CPU-based scikit-learn random forests and was lightweight relative to LLM inference and environment rollouts. GPT-5.4 was accessed through the public OpenAI API. Aggregate token counts and total GPU-hours were not logged.

C.9 ScienceWorld failure boundary

The ScienceWorld failure label is $y_{\text{fail}}(h) = \mathbf{1}[U_{\text{continue}}(h) \leq 0]$, the point at which continuing is no better than quitting; on the 364 pooled test prefixes, 132 (36.3%) satisfy it. Relabeling with $U_{\text{continue}}(h) \leq \delta$ for $\delta \in \{-0.10, -0.05, 0, 0.05, 0.10\}$, refitting the failure model, and reselecting its routing rule on validation gives failure-triggered regret 0.253, 0.249, 0.245, 0.242, and 0.239; the prefix-feature controller, which does not use the binary target, stays at 0.169.

C.10 Closed-loop and budget-suite tables

Tables 5 and 6 give the full numbers behind Figures 4 and 5.

Table 5: Closed-loop ALFWorld with a fixed stronger-model handoff; rates pooled over 300 episodes (50 per run), utility includes failure, handoff, and step costs. Failure is terminal failure. Quit at first decision is an analytical outside option: the controller acts after each base-agent step, so quitting at the first decision costs one step.

Policy	Utility	Success	Failure	Quit	Handoff	Steps
Continue only	-0.719	0.240	0.760	0.000	0.000	19.87
Always hand off	-0.391	0.400	0.600	0.000	1.000	14.13
Quit at first decision	-0.010	0.000	0.000	1.000	0.000	1
Failure-triggered routing	-0.146	0.373	0.360	0.267	0.480	13.57
Compact summary	-0.057	0.420	0.307	0.273	0.400	15.08
Prefix-feature controller	**0.020**	**0.450**	**0.267**	0.283	0.350	14.63

Table 6: Limited-budget branching in the four-action suite (Qwen2.5-7B and Llama-3.1-8B, six seeds each, 12 runs, 20/10/20 splits, 240 training and 240 test prefixes). Budget is the fraction of the 480 additional intervention branches acquired. Each action is the oracle choice on 31%, 11%, 35%, and 23% of held-out prefixes. Targeted acquisition adds batches of twelve prefix-action pairs with the largest across-tree variance in predicted utility. Absolute regret is not comparable to Table 1: the action set, utilities, and oracle differ.

Controller	Extra branches	Budget	Test regret	Gain retained
Failure-triggered, fixed GPT-5.4 handoff	0	0%	0.337	
Shared $Q(h, a)$, uniform	48	10%	0.194	61%
Shared $Q(h, a)$, targeted	48	10%	0.158	76%
Shared $Q(h, a)$, uniform	120	25%	0.151	79%
Shared $Q(h, a)$, targeted	120	25%	0.121	92%
Shared $Q(h, a)$, full branching	480	100%	0.101	100%

C.11 Protocol map

The ALFWorld regret values in this paper come from the following protocols and are comparable only within a protocol. Table 1 and Table 3: the main suite (26 ALFWorld runs, 16 ScienceWorld runs, 6 GSM8K runs, 6 HotpotQA runs), oracle defer-to-expert on ALFWorld, per-run training with validation selection and run-level aggregation; the intervention-aligned rows of Table 1 use the same runs and splits with predictors fit per run. Table 2, main-suite columns: the same runs and splits, one estimator and one router for every target (Section D.1); mid-episode columns and Table 8: the mid-episode prefix suite, eight separate runs with six prefixes per trajectory (Section D.2). Table 7: the 30 degraded-expert runs, under the protocol of Table 2. Table 11: the four eight-prefix repair probes. Table 12: the oracle probe of Section D.7. Table 14: the 84 regimes of Section D.9. Table 5: 300 fresh ALFWorld tasks, six model-seed runs, GPT-5.4 handoff, frozen controllers. Table 6: the separate four-action ALFWorld suite (12 runs, 20/10/20 splits). Table 10: the 26 main-suite runs with rows pooled across runs. Table 13: the cost-sensitivity sweeps of Section D.8. Section C.7: 72 test prefixes per interactive benchmark. Tables 9, 15, 16, and 17: pooled protocols that concatenate rows across runs, as stated in each caption.

D Additional Results

D.1 Matched-target ablation

The main-suite columns of Table 2 use the 26 ALFWorld main-suite runs and their splits; the mid-episode columns are described in Section D.2. For every target the input is the base prefix feature vector with self-reported confidence (26 dimensions), the estimator is the random-forest regressor of Section C.5 (200 trees, depth 6, minimum leaf 2, fixed random seed) with binary labels fit as 0/1 targets, and the router is a two-threshold rule on the predicted scalar. Thresholds are taken from the same 12-point quantile grid as the deployed failure pipeline; thresholds and the assignment of the three actions to the three regions are selected together on the run's validation split by validation utility, so every target may use any action order. Regret is evaluated on the run's test rows and averaged over runs. The two-quantity row fits the continuation and intervention utility regressors and routes with the value rule of the main pipeline (margin and quit floor selected on validation). Its regret of 0.069 is

below the 0.110 of Table 1 with the same features because Table 1 selects the controller family on validation splits of 4 to 20 prefixes, whereas this row fixes the value family; Section D.13 reports the same pattern for a fixed conservative controller.

The deployed failure pipeline fits the failure label with a classifier and fixes the order continue, intervene, quit; it reproduces 0.506 on the main-suite rows and 0.286 on the mid-episode suite. Fitting the two binary labels with that classifier instead of the regressor gives 0.494 for the failure label (0.500 and 0.525 under two further seeds) and 0.172 for intervention success (0.131 and 0.146), with the utility rows unchanged, run-level intervals $[0.235, 0.407]$, $[0.253, 0.434]$, $[0.299, 0.512]$, and $[0.352, 0.503]$ for the four intervention-side rows, and 23 to 25 of 26 runs improved. Under two further seeds of the regressor version the means are 0.507 and 0.519 for the failure label, 0.109 and 0.089 for intervention success, 0.144 and 0.157 for the advantage, 0.089 and 0.095 for intervention utility, and 0.074 and 0.062 for the two-quantity router. The selected action orders are informative: the failure label keeps the deployed order in most runs, while every intervention-side target places quitting in the low-score region, so a low predicted intervention value routes to quit rather than to continuation. Task-clustered 95% intervals (Section C.6) for the regressor version are $[0.339, 0.424]$ for intervention success, $[0.308, 0.395]$ for the advantage, $[0.368, 0.443]$ for intervention utility, and $[0.367, 0.448]$ for the two-quantity router.

Table 7 repeats the ablation on the 30 degraded-expert runs of the intervention-quality ladder (six runs per variant, 16 test prefixes each, so individual cells are noisy). With a noisy expert the picture of Table 2 persists at every noise level: all four intervention-side rows stay well below the failure label, and their paired intervals over the six runs all exclude zero; the weakest case is intervention utility at noise 0.40, $[0.017, 0.466]$ with four of six runs improved. With a truncated expert that executes two or three steps and stops, failure routing itself has lower regret, the single intervention-side scalars are worse than the failure label on average with intervals covering zero, and only the two-value pair still improves on it, by 0.126 $[0.033, 0.220]$ (four of six runs) and 0.133 $[0.018, 0.239]$ (three of six). Truncation changes both how long the intervention acts and how the episode ends, so it is not only a weaker expert; in these runs a single intervention-side scalar removes most of the regret while the expert is capable, and only the pair still improves on the failure label under truncation.

Table 7: Matched-target ablation across degraded experts on ALFWorld (same protocol as Table 2 with the binary labels fit by a classifier; run-level mean regret; six runs per variant). Noise variants replace each expert step by a random admissible non-expert command with the stated probability; truncated variants execute two or three expert steps and stop.

Expert variant	Failure	Cont. util.	Interv. succ.	Advantage	Interv. util.	Both
Noise 0.10	0.609	0.383	0.283	0.152	0.124	0.099
Noise 0.25	0.605	0.380	0.174	0.133	0.058	0.099
Noise 0.40	0.597	0.380	0.185	0.222	0.350	0.126
Truncated, two steps	0.232	0.365	0.358	0.304	0.316	0.106
Truncated, three steps	0.347	0.366	0.380	0.381	0.434	0.214
All 30 runs	0.478	0.375	0.276	0.238	0.256	0.129

D.2 Mid-episode prefix suite

To test the same comparisons at decision points spread over the episode, we collected a second ALFWorld suite with the same base agents, task types, split rule, expert intervention, costs, and 20-step budget, but with six decision prefixes per base trajectory at evenly spaced steps (steps 0, 4, 8, 11, 15, and 19 of a 20-step trajectory): eight runs (Qwen2.5-7B and Llama-3.1-8B, seeds 13, 17, 29, and 37), 20/10/20 tasks per run, which gives 115 to 120 training, 57 to 60 validation, and 116 to 120 test prefixes per run against 4 to 20 validation prefixes in the main suite; 952 held-out test prefixes over 65 distinct tasks in total. Mid-episode the realized-best action is intervene on 78%, quit on 13%, and continue on 9% of prefixes, so intervening at every prefix is a strong constant reference. Table 8 reports the deployed pipelines and the matched-target ablation under the protocol of Table 2; the matched rows also form the right half of Table 2. Failure-triggered routing does no better than the constant reference. Intervention success, intervention utility, and the two-value pair improve on the failure label in all eight runs; the advantage improves in seven and is the weakest of them, as in the main suite. The regret reduction of the validation-selected controller over failure-triggered routing is

0.199, with a run-level interval of $[0.171, 0.225]$ and a task-clustered interval of $[0.162, 0.228]$. With the binary labels fit by a classifier instead, the matched rows are 0.275, 0.245, 0.087, 0.148, 0.098, and 0.095.

Table 8: Mid-episode prefix suite on ALFWorld (eight runs, six prefixes per trajectory; run-level mean regret; intervals are paired bootstraps over runs). Matched rows use the protocol of Table 2: same features, regressor, and validation-selected two-threshold router; only the label changes.

Policy or supervision target	Regret	Reduction [95% CI]	Runs
Failure-triggered routing (deployed pipeline)	0.286	—	—
Intervene at every prefix	0.263	0.023	—
Action-conditioned controller, validation-selected family	0.088	0.199 $[0.171, 0.225]$	8/8
Matched: failure label	0.295	—	—
Matched: continuation utility	0.245	0.049 $[-0.005, 0.110]$	5/8
Matched: intervention success	0.081	0.214 $[0.178, 0.257]$	8/8
Matched: advantage $U_i - U_{\text{continue}}$	0.148	0.147 $[0.077, 0.222]$	7/8
Matched: intervention utility	0.098	0.196 $[0.165, 0.241]$	8/8
Matched: both values, pair router	0.095	0.199 $[0.152, 0.255]$	8/8

D.3 How far can better scalars go?

Recalibrating the same scalar changes its values but not the threshold rules it can express (Remark 1; measured in Section D.5). A different question is whether choosing a *different* scalar, one more aligned with intervention advantage, would close the gap without requiring action-conditioned control. Within each run, we select a scalar and controller variant by validation utility from eight candidates: failure score, self-reported confidence, predicted continuation and intervention correctness, and the expected-utility and direct-value predictions for each action (see Section C.5). We then average held-out test regret across the 26 ALFWorld runs. This auxiliary search uses the action-conditioned controller family, a broader router search than the single-scalar pipeline in Table 1.

Under this per-run protocol, the validation-selected single scalar reaches 0.152 against 0.506 for failure-triggered routing and 0.110 for the prefix-feature controller. Intervention correctness is selected in 16 runs, continuation value in five, intervention value in three, confidence in one, and failure score in one.

Intervention-aligned candidates recover much of the gain, with a remaining mean regret gap of 0.043 to the prefix-feature controller. Selection varies across runs; the matched comparison in Section D.1 isolates the effect of the supervision target.

D.4 Compact multi-scalar routing

A complementary question is how well a small combination of scalars supports control. This subsection uses pooled fitting; Section D.3 and the compact-summary row of Table 1 use per-run fitting and selection. Their regret values are not directly comparable.

Protocol. We pool train, validation, and test rows separately across all runs within each benchmark. For each run, per-row scalar features are computed using models fit on that run's training split; the resulting scalar vectors are concatenated across runs. We then train multi-class oracle-action classifiers on all $\binom{8}{2} + \binom{8}{3} = 84$ two- and three-scalar combinations from the eight candidate scalars, using both logistic regression (with balanced class weights) and random forest (200 trees, max depth 6, balanced class weights). The best combination is selected on validation utility and evaluated on held-out test data. Because this searches over many combinations, the result is an exploratory validation-selected outcome, not a bound.

Note on train-set scalars. The scalar features for the training split are in-sample predictions from the per-run random-forest models (the same models that produce the val/test scalars). This creates a distributional mismatch between train and val/test features whose direction of effect has not been established; we did not run a cross-fit comparison.

Table 9: Compact multi-scalar routing under the pooled protocol. Gap closed is N/A when the prefix-feature controller does not improve on the baseline. "Single failure" is logistic regression on failure score alone; "best multi-scalar" is the val-selected combination. "Gap closed" is computed relative to the single-failure → prefix-feature-controller gap within this protocol. All numbers are pooled test regret (lower is better).

Benchmark	Single failure	Best multi-scalar	Prefix-feature controller	Gap closed
ALFWorld	0.451	0.057	0.015	90%
ScienceWorld	0.255	0.217	0.149	36%
GSM8K	0.469	0.476	0.369	−8%
HotpotQA	0.434	0.416	0.442	N/A

On ALFWorld, the best three-scalar combination (failure score, predicted intervention correctness, and predicted intervention expected utility) reduces pooled regret from 0.451 for single-failure-score routing to 0.057, compared with 0.015 for the prefix-feature controller, closing about 90% of the gap under this pooled protocol. Logistic regression (0.067) and random forest (0.057) reach similar test regret with the validation-selected combination; this compares two routers and does not isolate the target. The best-performing ALFWorld combinations all include predicted intervention correctness or predicted intervention expected utility, and the failure-score pipeline has higher regret in this comparison. On ScienceWorld the compact summary closes 36% of the gap, consistent with its smaller gain in Section 5.5. On GSM8K the compact summary does not improve under the deployed verify intervention; on HotpotQA it improves slightly on the single-failure baseline while the prefix-feature controller does not, so no positive gap is defined (Table 9). The oracle probe in Section D.7 is consistent with weak intervention quality limiting GSM8K and with the failure score carrying more of the decision on HotpotQA. This supports the paper's reading: the deficit lies in summaries that track failure risk and omit intervention value, not in the use of a single scalar as such.

D.5 Calibration decomposition

Recalibrating a fixed score with Platt scaling and reselecting thresholds on validation improves prediction metrics while the measured control regret of threshold routing is unchanged; isotonic regression creates ties and, on the failure score, worsens both prediction metrics and regret. Remark 1 concerns the policy class, not the finite-sample reselection; the equality is a measurement. Values are computed on the main-suite rows pooled across runs and are not comparable to the per-run results of Table 1.

Table 10: Calibration decomposition on ALFWorld, evaluated on the main-suite rows pooled across runs, with thresholds reselected on validation after each transform. Platt scaling improves ECE and Brier while measured control regret is unchanged; isotonic regression on the failure score worsens all three.

Scalar	Method	ECE	Brier	Control regret
Confidence	Raw	0.463	0.397	0.318
	Platt	0.006	0.110	0.318
	Isotonic	0.022	0.110	0.318
Failure score	Raw	0.060	0.096	0.358
	Platt	0.047	0.094	0.358
	Isotonic	0.097	0.103	0.462

D.6 Prompt-only same-model repair probes

Table 11 lists the ALFWorld prompt-only same-model repair probes summarized in Section 5.5. They rarely improve over continuation, and every validation-selected controller chose quit.

D.7 Oracle intervention probe

The main text uses an oracle intervention probe on GSM8K and HotpotQA; because the probe changes the intervention and the supervision and retrains the learners, its results are read as consistent with, not proof of, the two readings. The protocol is as follows: for each run, we replace every

Table 11: ALFWorld prompt-only same-model repair probes. In *reflect and retry*, the base model reflects on its proposed action and chooses a different one; in *replan*, it replans the next three steps before the episode continues; *plan, then replan* first writes a plan. These interventions are boundary evidence, not main positive results: they rarely improve over continuation, and deployable controllers collapse to `quit`.

Intervention	Rows	Repair succeeds	Beats continuation	Oracle repair	Oracle quit
Reflect and retry (Qwen)	8	1	0	0.000	0.750
Reflect and retry (Llama)	8	2	0	0.000	0.750
Replan, 3 steps (Qwen)	8	3	1	0.125	0.625
Plan, then replan, 3 steps (Qwen)	8	2	0	0.000	0.750

verify-branch outcome with that of a gold-answer oracle that always succeeds, so the verify branch is correct and its utility is $1 - c_a$ at every prefix. Continue-branch outcomes, features, and quit utilities are unchanged. We then retrain both the failure-triggered controller (a two-threshold rule on the failure score, thresholds and action order selected on validation) and the EU-LCB controller on this counterfactual data and evaluate on held-out test rows.

Table 12: Oracle intervention probe on the reasoning benchmarks. "Gain" is failure-triggered regret minus EU-LCB regret; higher gain means more room for action-conditioned control. Under oracle intervention, GSM8K's gain increases substantially, while HotpotQA's gain barely changes.

		Failure regret	EU-LCB regret	Gain
GSM8K	Original	0.414	0.386	0.028
	Oracle	0.144	0.018	0.127
HotpotQA	Original	0.454	0.411	0.043
	Oracle	0.072	0.020	0.052

The probe is a diagnostic, not a deployment result: it asks how the gain changes when intervention quality is removed as a constraint. It cannot reveal a sufficiency gap. With the verify branch always correct, $u_{\texttt{verify}} = 0.9$ for every row, and with wrong penalty 0.5 the continuation value at true failure probability p is $1 - 1.5p$, so the optimal rule is to intervene whenever $p > 1/15$ and never to quit: the true failure probability is sufficient under the probe. The gain that remains between the two learned pipelines therefore reflects how well each estimates and routes on that probability from prefix features, not information the failure score lacks. On GSM8K, the $4.5\times$ increase in gain is consistent with the deployed same-model verify intervention limiting the achievable gain; the probe changes the intervention and the supervision, so it does not measure a latent $\mathrm{Gap}(g)$. On HotpotQA, the near-unchanged gain ($1.2\times$) is consistent with the failure score carrying more of the decision in this regime.

D.8 Cost sensitivity on the interactive benchmarks

Table 13 reports the deployed gain under a 5×5 sweep over intervention cost and wrong-answer penalty, pooling the 56 ALFWorld runs of the main suite and the intervention-quality ladder and the 16 ScienceWorld runs. ALFWorld stays positive in all 25 cells and ScienceWorld in 23.

Table 13: Cost-sensitivity summary on the interactive benchmarks. Gains are failure-triggered regret minus prefix-only action-conditioned regret; positive values favor richer control.

Benchmark	Positive cells	Gain range	Gain along the cost line (c: 0 to 0.20)
ALFWorld	$25/25$	$[0.170, 0.345]$	$0.320 \to 0.252$
ScienceWorld	$23/25$	$[-0.008, 0.141]$	$0.106 \to -0.006$

D.9 A branched-data diagnostic for anticipating useful regimes

Section 2.2 states the conditions under which richer control pays off. Can these conditions be measured before committing to a more complex controller? We propose a development-time diagnostic

computed from branched data. The idea is simple: if a classifier trained on prefix features can predict the oracle action better than a classifier trained on the scalar alone, then the prefix carries control-relevant information that the scalar misses. We call this quantity *exploitability beyond scalar*. If $\widehat{A}^*_g$ denotes the oracle-action predictor trained on scalar g alone and $\widehat{A}^*_{g,x}$ the predictor trained on scalar plus prefix features $x(h)$, then

$$\mathrm{Exploit}(g) = \mathrm{Score}\Big(\widehat{A}^*_{g,x}\Big) - \mathrm{Score}\Big(\widehat{A}^*_g\Big),$$

where $\mathrm{Score}(\cdot)$ is measured out of sample on held-out prefixes. In the main experiments we use utility-weighted held-out prediction quality, so higher exploitability means that prefix information beyond the scalar makes oracle-action prediction easier.

The main empirical pattern appears in Figure 7. Across 84 regimes, exploitability correlates with deployable gain (overall Pearson $r = 0.716$), and the relationship remains visible within ALFWorld itself ($r = 0.604$). ALFWorld has the largest exploitability (0.310) and the largest gain (0.327); the other three benchmarks are much lower on both.

Interpretation and scope. Exploitability shares the same prefix feature space as the downstream prefix-feature controller, so part of the correlation is mechanical: both quantities reflect how much control-relevant information the prefix carries beyond the scalar. We therefore read exploitability as a development-time diagnostic, a check from branched evaluation data on whether richer control is likely to help in a regime, and not as a model-agnostic causal predictor. To test whether the association is merely mechanical, we evaluate cross-model transfer: compute exploitability on one agent model's data and measure gain on a different agent model's data, using 13 matched ALFWorld seed/split pairs of Qwen2.5-7B and Llama-3.1-8B. Exploitability computed on Qwen2.5-7B data predicts Llama-3.1-8B gains ($r = 0.752$), and Llama-3.1-8B exploitability predicts Qwen2.5-7B gains ($r = 0.563$), both comparable to within-model correlations (0.761 and 0.851). The association is observed across model families; shared tasks, labels, estimators, and selection are not excluded as sources of the correlation, and this is not a prospective held-out-family validation. To check that the association is not unique to $g = p_{\mathrm{fail}}$, we repeated the analysis with self-reported confidence as the candidate scalar; the correlation drops to $r = 0.339$ overall and to $r = 0.087$ within ALFWorld (Table 14). Dependence-robust checks remain positive (Section D.10).

D.10 Dependence-robust view of the exploitability diagnostic

Figure 7 uses one point per run. It does *not* include the cost-sweep cells or alternate-scalar variants as additional points; those analyses are reported separately in Section 5.5 and Table 14. To check sensitivity to grouping, we aggregated the 84 run-level points into 12 protocol groups: six ALFWorld intervention variants, one ScienceWorld suite, two GSM8K base-model groups, and three HotpotQA seed groups. We then recomputed the correlation between exploitability and deployable gain. The group-level correlation remains strong ($r = 0.962$). Leaving out one group at a time keeps the run-level correlation in the range 0.677–0.735. These checks do not remove all dependence concerns, but they show that the positive association in Figure 7 is not driven by any single group or by the cost/multi-scalar sweeps.

D.11 Alternative scalar candidates

Table 14 compares the main failure-score scalar to self-reported confidence on the same 84 regimes used in the main exploitability analysis. Both signals are informative; the diagnostic built on the failure score correlates more strongly with gain on these regimes, which does not by itself rank the two scores' control value. This is consistent with the paper's main thesis. The question is not whether one can find a scalar that correlates with success at all, but whether a candidate scalar preserves enough information for intervention control.

D.12 Per-variant action-conditioned estimator results

Table 15 reports control regret for several action-conditioned estimator variants across benchmarks. These tables answer a different question from Table 3. Table 3 is the main result, in which failure-triggered routing and the prefix-feature controller see the same prefix-only information. By contrast,

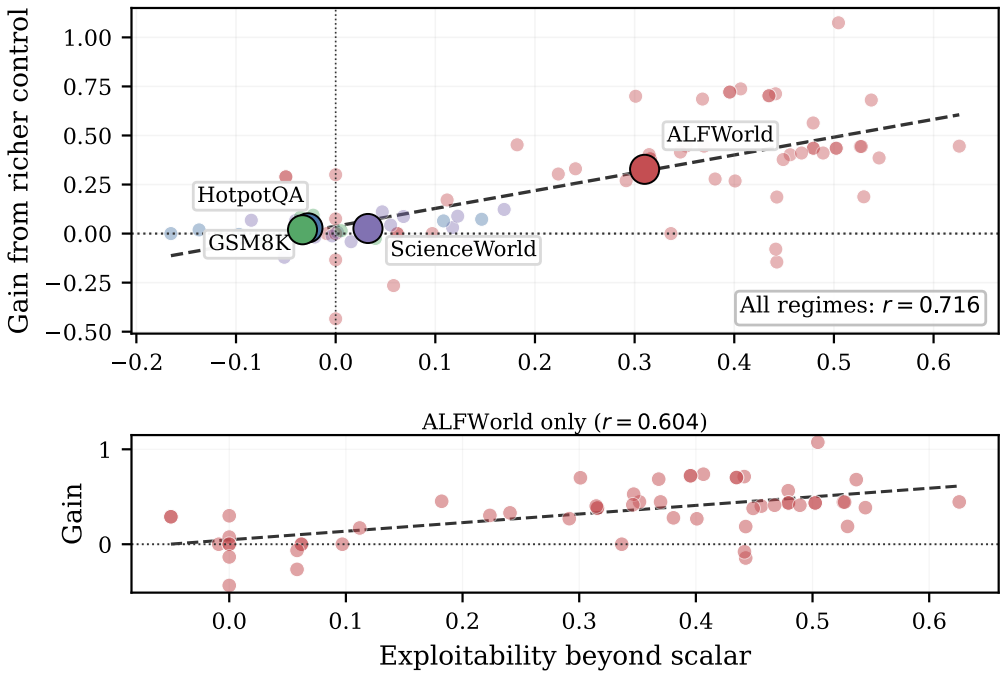

Figure 7: Exploitability beyond scalar predicts the gain from richer control. Each point is one of 84 regimes. The horizontal axis is the held-out predictive improvement from scalar-only to scalar-plus-prefix prediction of the oracle action; the vertical axis is the regret reduction of the prefix-feature controller over failure-triggered routing. The lower panel shows the same relationship within ALFWorld alone.

Table 14: Exploitability diagnostics for two candidate scalars on the same 84 regimes. Higher correlation means the measured mismatch better predicts deployable action-conditioned gain.

Candidate scalar	Overall correlation	ALFWorld correlation
Failure score $p_{\text{fail}}(h)$	0.716	0.604
Self-reported confidence	0.339	0.087

the tables below summarize supplementary experiments that compare estimator variants on the pooled benchmark suites, including auxiliary preview-aware variants used only for supplementary analysis. Their absolute regret values should therefore be read as *within-appendix estimator ablations*, not as directly comparable replacements for the prefix-only numbers in Table 3. The *value* variant directly predicts per-action utility, *explicit-EU* decomposes estimation into correctness probability and known reward structure, *EU-LCB* adds a lower confidence bound to explicit-EU, and *two-stage* first decides whether to intervene then selects among interventions.

Table 15: Control regret by action-conditioned estimator variant in the supplementary estimator comparison (pooled across seeds and models; lower is better). The best variant per benchmark is bolded. These values are within-appendix ablations and are not directly comparable to the prefix-only main results in Table 3.

Variant	GSM8K	HotpotQA	ALFWorld	ScienceWorld
Failure-triggered	0.423	0.436	0.491	0.245
Value (direct utility)	0.377	0.424	0.050	**0.178**
Explicit-EU ($\beta = 0$)	**0.364**	0.404	**0.010**	0.206
EU-LCB	0.395	**0.358**	**0.010**	0.182
Two-stage	0.407	0.403	0.146	0.191

Several patterns emerge. First, on ALFWorld all action-conditioned variants outperform failure-triggered routing under this supplementary protocol, which includes auxiliary preview-aware variants; this does not identify the source of the main prefix-only gain. Second, the best variant is benchmark-

dependent: explicit-EU dominates on GSM8K, EU-LCB is strongest on HotpotQA, explicit-EU and EU-LCB are tied on ALFWorld at the displayed precision, and the direct value estimator is best on ScienceWorld. The ordering among estimators is reported as observed; differences in means do not identify bias or variance contributions. Third, all four variants improve on failure-triggered routing on ScienceWorld, with the direct value estimator performing best in this supplementary comparison. Finally, the two-stage variant has higher regret than the direct value estimator on ALFWorld (0.146 vs. 0.050) in this suite.

D.13 Fixed conservative controller versus validation-selected family member

The main results (Table 1) select among controller families on validation, per run. This section compares that choice with a fixed conservative controller that fixes the family and tunes only its conservatism parameter β on validation, using the pooled data of the supplementary estimator comparison (Section D.12). Table 16 compares the two.

Table 16: Fixed conservative controller versus validation-selected best action-conditioned family member in the supplementary estimator comparison (control regret, pooled). Val-selected picks the family member with highest validation utility per run, then evaluates on test. These numbers should be read relative to Table 15, not as direct replacements for Table 3.

Controller	GSM8K	HotpotQA	ALFWorld	ScienceWorld
EU-LCB	0.395	0.358	0.010	0.182
Val-selected	0.405	0.397	0.063	0.184

The fixed conservative controller outperforms the validation-selected approach on all four benchmarks, though the margin on ScienceWorld is negligible (0.182 vs. 0.184). This is consistent with sensitivity to validation selection, which the fixed recipe avoids by tuning a single parameter.

D.14 WebShop diagnostic

WebShop [Yao et al., 2022] is included as a supporting diagnostic only, not as a main benchmark, because it has fewer runs ($n = 2$–3) and different action semantics. On WebShop, the measured diagnostic is small (0.033–0.050); a weaker violation of the sufficiency condition is one possible explanation. Consistent with this, the validation-selected action-conditioned family member outperforms failure-triggered routing, but the fixed conservative controller does not always improve over failure-triggered routing on this benchmark. A bias-variance explanation is possible but not identified by these runs.

D.15 Cross-model transfer

In the main experiments, controllers are trained and evaluated on trajectories from the same agent model (within-model regime). A natural question is whether controllers trained on one model's trajectories generalize to another. We test this by training on Qwen2.5-7B trajectories and evaluating on Llama-3.1-8B trajectories (and vice versa), across all available seed pairings.

Table 17: Cross-model transfer versus matched-subset within-model control regret (pooled; lower is better). Transfer pairs train on one model and evaluate on the other, so the within-model column reports the corresponding matched within-model subset rather than the full suite used in the main text.

	Within-model		Cross-model transfer	
Benchmark	Failure	EU-LCB	Failure	EU-LCB
GSM8K ($n = 6$)	0.423	0.395	0.440	0.405
HotpotQA ($n = 2$)	0.436	0.358	0.464	0.337
ALFWorld ($n = 6$)	0.716	0.005	0.574	0.005
ScienceWorld ($n = 4$)	0.270	0.183	0.312	0.191

Table 17 shows that cross-model transfer barely degrades conservative action-conditioned performance on the matched subset. On all four benchmarks, the transfer regret of the EU-LCB controller

is within 0.05 of the within-model regret, and on ALFWorld the regret is near-zero in both settings. The relative ordering between failure-triggered routing and the EU-LCB controller is preserved under transfer on all benchmarks for the subset and estimator of this table; task repetition, features, protocol, and selection are not excluded as contributing factors.

D.16 Scaling curves

Figure 8 reports control regret as a function of training set size on ALFWorld, ScienceWorld, and GSM8K. The pattern is consistent with the selection-noise discussion in Section 4.2: the EU-LCB controller overtakes failure-triggered routing earlier on ALFWorld than on ScienceWorld.

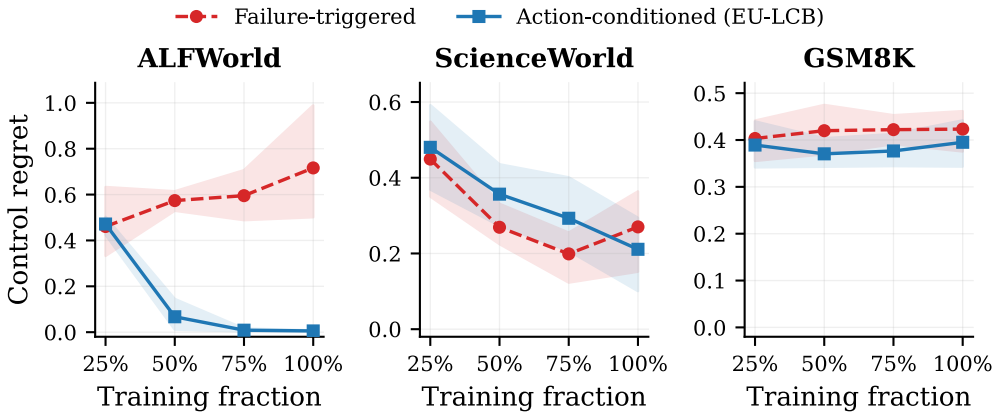

Figure 8: Control regret versus training set size. Lines are run-level means; bands are 95% bootstrap intervals over runs. The EU-LCB controller overtakes failure-triggered routing earlier on ALFWorld than on ScienceWorld.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction state the sufficiency criterion (Section 2), the ladder and matched-target comparisons (Sections 5.1 and 5.2), the closed-loop result (Section 5.4), and the benchmarks where gains are small (Section 5.5); each number quoted there appears in the corresponding table or figure.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The Limitations paragraph in Section 7 lists the in-sample reference, the missing interval against the quit reference online, the probe's change of supervision, the budget suite's action set, and the concentration of strong results on ALFWorld.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: Definition 1, Proposition 1, Remark 1, and the abstraction loss are stated in Section 2; all proofs and the lower bound (Proposition 3) are in Appendix A.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: Branching protocol, splits, utilities, intervention definitions, features, estimators, routers, and statistical procedures are specified in Appendix C; a per-table protocol map is given in Appendix C.11.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in

some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: Code for branch collection and controller evaluation is available at `https://github.com/bennidict23/calibration-is-not-control`, with instructions to collect branched ALFWorld data and rerun the controller comparison. The collected branched data are not released.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Section 4 and Appendix C give benchmarks, agent models, intervention definitions, utility parameters, split sizes, feature lists, estimator hyperparameters, and router search grids.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: All main comparisons report paired-bootstrap 95% intervals over runs (Appendix C.6); the main ALFWorld comparison additionally reports a task-clustered interval over the 84 held-out tasks.

   Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Appendix C.8 gives the hardware (RTX A5000 GPUs for local inference, CPU for controller training), the API used for GPT-5.4, the measured collection cost of branching, and the controller's decision time; aggregate GPU-hours were not logged.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research conforms with the NeurIPS Code of Ethics; it uses public benchmarks and models and involves no human subjects or personal data.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes]

Justification: Section 7 states the practical implication: oversight signals should be judged by the decisions they support. As with other oversight methods, more reliable intervention decisions could also make a misused agent more reliable; the work adds no capability beyond the base agent and the intervention.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: No models or datasets with a high risk of misuse are released; the release contains code only.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: ALFWorld, ScienceWorld, GSM8K, HotpotQA, Qwen2.5-7B-Instruct, Llama-3.1-8B-Instruct, and GPT-5.4 are cited in Sections 4 and 5.4; we use them under their licenses and terms of use.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `https://paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: The new asset is the code for branch collection and controller evaluation, released with a README that documents installation and usage.

Guidelines:

- The answer [N/A] means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [N/A]

Justification: The paper involves no crowdsourcing and no research with human subjects.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [N/A]

Justification: The paper involves no research with human subjects.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [Yes]

Justification: LLMs are the base agents and the handoff model whose oversight is studied (Section 4); this is the core methodology and is described throughout. Generative AI tools were also used to assist with drafting and polishing the manuscript and with literature retrieval; all such content was reviewed and verified by the authors.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.