DOOMED FROM THE START: EARLY ABORT OF LLM AGENT EPISODES VIA A RECALL-CONTROLLED PROBE CASCADE

Anonymous authors
Paper under double-blind review

ABSTRACT

Large language model (LLM) agents often waste inference compute by continuing multi-step trajectories that are already doomed to fail. We study early failure prediction and inference-time early stopping for LLM agents using hidden-state probes. Lightweight linear probes on internal activations predict eventual task failure from the first interaction round, substantially earlier than agent-monitoring methods based only on observable behavior. We turn this signal into a recall-controlled abort cascade for reducing LLM agent inference costs. The cascade applies a distribution-free calibrated failure detector at each early interaction round and jointly optimizes per-round recall budgets. This design ensures that eventually successful episodes survive all early-stopping gates at a user-specified global recall rate. After selection, the cascade is frozen and certified on independent data, providing an exact post-selection recall guarantee. We evaluate the method on TextCraft and WebShop with Qwen-2.5-7B, Llama-3.2-3B, and Qwen3-1.7B. The proposed LLM agent early-stopping cascade outperforms the best single-gate baseline in every model-environment pair, saving 1.5–8.8$\times$ more compute at a 90% recall target. Achieved recall remains within one standard deviation of its target in all 24 configurations. The strongest settings reduce generated tokens by 60.2% on TextCraft and 54.9% on WebShop at 90% recall, while retaining savings of 45.0% and 41.5% at 95% recall. Behavior-only monitoring is consistently weaker, and adding behavioral features to hidden-state probes provides no further gain. We also characterize the sample complexity required to certify high-recall early-stopping policies. The code will be released soon.

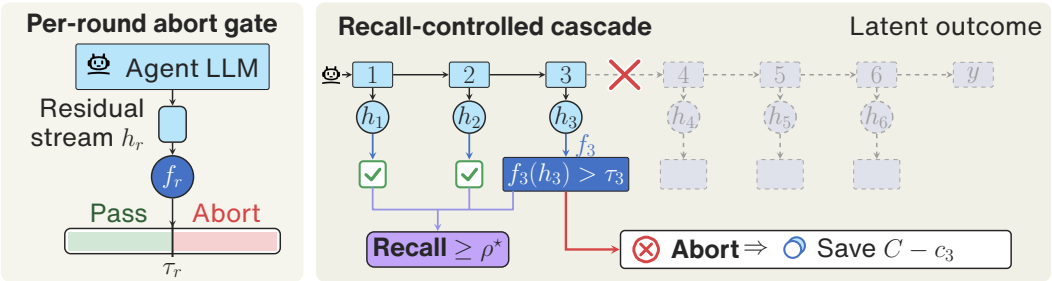

Figure 1: Recall-controlled abort cascade. At each of the first R_g rounds, a linear probe f_r reads the agent's hidden state h_r and a gate aborts the episode if the failure score exceeds a calibrated threshold τ_r; an episode completes only if it survives every gate. Thresholds carry per-round distribution-free recall guarantees, and the per-round budgets t_r are jointly searched so that the *episode-level* success recall meets a user-chosen target $\rho^\star$; the selected cascade can then be certified on independent held-out data, yielding an exact post-selection guarantee.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

1 INTRODUCTION

LLM-based agents increasingly tackle long-horizon tasks such as web navigation, tool use, and embodied simulation, where a single episode spans many rounds of interaction and consumes thousands of generated tokens. A large fraction of this compute is wasted: when an agent has misunderstood the task, entered an unrecoverable state, or begun to loop, the episode is already doomed long before it formally times out or returns a wrong answer. If we could detect such doomed episodes early and abort them, the saved compute could be reallocated to retries, sampling additional trajectories, or simply reducing serving cost.

Three obstacles stand in the way. First, we need a signal that distinguishes doomed episodes from eventually-successful ones *early* in the trajectory, and early is precisely when behavioral evidence is scarcest. We find that scorers reading only the agent's observable behavior are barely better than chance in the first round and become informative only around rounds 3–4, by which time over a third of episodes have already finished and much of the useful remaining compute is gone. Lightweight probes on the agent's internal activations show the opposite pattern: at the very first round they already match or exceed the surface scorer's eventual peak, attained only two to three rounds later, and they reach their own peak at round 2 (Figure 3).

Second, any abort policy is only usable in deployment if it comes with a controllable guarantee on the harm it causes: aborting an episode that would have succeeded silently destroys task reward, so practitioners need a predeployment bound on the rate of such false aborts. Third, because agent episodes are sequential, a monitor that re-evaluates the episode at every round faces accumulating risk. Even if each individual check rarely kills a good episode, a successful trajectory must survive *all* of them, so per-round guarantees do not compose into the episode-level guarantee that matters. For example, if two gates each pass 98% of successful episodes, the cascade passes 98% when they reject the same successes but only 96% when their false-abort sets are disjoint. Marginal per-round rates therefore do not determine global recall.

We address these with a recall-controlled *cascade*, shown in Figure 1. At each of an episode's first R_g rounds, a gate aborts the episode if its probe score exceeds a threshold τ_r. Each threshold is calibrated so that an exact binomial (Clopper–Pearson) lower confidence bound on the gate's survival rate for successful episodes meets a per-round recall budget t_r, and the budget vector $(t_1, \ldots, t_{R_g})$ is searched on a disjoint validation split to maximize compute savings subject to the *global* recall (the fraction of eventually-successful episodes that survive every gate) meeting a user-chosen target. A safety margin makes the search robust to validation noise. When a formal global guarantee is required, the selected cascade is frozen and evaluated once on an independent certification sample; because the cascade is fixed before these data are seen, a single Clopper–Pearson lower bound on that sample yields an exact, distribution-free guarantee that is immune to the preceding search, no matter how many candidates the search considered.

The cascade is strongly compute-positive across the entire evaluation matrix. At a 90% global recall target, the strongest TextCraft and WebShop cells save 60.2% and 54.9% of generated tokens, $1.5\times$ and $6.6\times$ the best *single-gate* policy at the same target; across all six cells the cascade's advantage over the best single gate ranges from $1.5\times$ to $8.8\times$, confirming that much of the practical value comes from distributing recall budget across rounds. At a conservative 95% target the cascade still saves up to 45.0% (TextCraft) and 41.5% (WebShop), and its achieved recall stays within one standard deviation of its target in all 24 evaluated configurations. Just as important, when the signal is weak, the cascade degrades gracefully toward a conservative near-no-op policy that prioritizes successful episodes, the appropriate failure mode for a deployed monitor. Holding the scorer fixed, the allocation comparison shows that recall budgets should track when each cell becomes predictable; a separate ablation attributes the early signal itself to the activations: behavior-only monitoring is consistently weaker, and stacking behavioral features onto the probe adds nothing.

Finally, we give an honest account of what *certified* recall control costs in data. With n successful episodes in an independent certification sample, a one-sided certificate at confidence 95% can support recall targets only up to $0.05^{1/n}$: 114 successes cap certifiable targets near 0.974, while targets of 0.98 and 0.99 require roughly 149 and 299 successful episodes (Figure 2). The certificate makes this data requirement explicit: the same machinery that saves compute also tells the practitioner, before deployment, which recall promises the available data support.

Our contributions are:

- We demonstrate across a full 2×3 matrix that eventual task failure of an LLM agent is predictable from internal activations within the first interaction rounds, before low-cost behavioral features become comparably informative.
- We propose, to our knowledge, the first abort policy for LLM agents that targets *episode-level* success recall across multiple sequential decision points, via per-round distribution-free gates whose recall budgets are jointly optimized under a global constraint, together with an exact, distribution-free post-selection certificate on independent data.
- We show the cascade saves up to 60.2% of inference compute at 90% global recall, outperforming the best single-gate policy in every cell, characterize the data cost of certifying stricter targets, and use ALFWorld as a stress test showing that the method fails conservatively and preserves recall when the signal is weak.

2 METHOD

2.1 PROBLEM SETUP

An agent episode is a sequence of interaction rounds $(s_1, a_1, s_2, a_2, \dots)$ between an LLM policy and an environment, terminating with a binary outcome $y \in \{0, 1\}$. We index rounds from one; episodes run up to $R_{\text{full}} = 20$ rounds. Let c_r denote the cumulative inference cost incurred through round r and C the total cost of the full episode.

An *abort cascade* places a gate at each of the first R_g rounds. We use $R_g = 6$ throughout. At gate r, every episode still running is scored by a per-round scorer f_r and aborted if $f_r(x) > \tau_r$, losing any potential success but saving the remaining cost $C - c_r$. Episodes terminate at different rounds; in our data, over a third of episodes finish within the first two rounds (see supplementary alive-episode curves), so a late gate sees only the episodes still alive there: late gates incur less recall risk and guard less remaining compute. This trade-off is exactly what the cascade optimizes.

Aborting an episode that would have succeeded is the critical error mode. We target *global success recall*: among episodes with $y = 1$, the fraction that survive *every* gate and run to completion. The design goal is to maximize expected cost savings subject to a user-specified global recall floor $\rho^\star$. Note that controlling the recall of each gate in isolation does not control the global recall: false-abort probability accumulates across gates, and the accumulation depends on how many successes each gate sees.

2.2 PER-ROUND FAILURE SCORERS

At each gate round r, we extract a feature vector $x \in \mathbb{R}^d$ from the agent LLM's internal activations: the residual-stream hidden state at the final token of the agent's generated action in round r, at a single layer selected once per matrix cell by a preliminary per-layer probe-AUC sweep. In our experiments this vector is recovered by a teacher-forced forward pass over the logged trajectory. Online deployment can instead collect selected-layer hidden states through an instrumented serving stack during inference (see Discussion), avoiding offline replay. Layer choice is fixed before any recall-target or cascade evaluation and remains fixed across targets. A per-round probe f_r, a logistic regression on standardized features with L_2 regularization and $C = 1$, is trained to predict eventual failure $1 - y$ from episodes alive at round r.

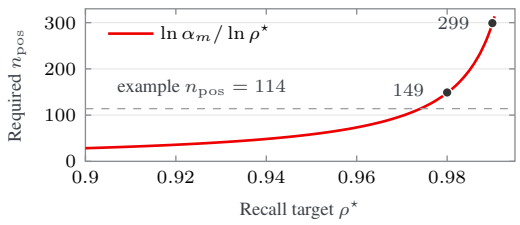

Figure 2: Sample complexity of the global recall certificate ($\alpha_m = 0.05$): minimum successful certification episodes. With $n_{\text{pos}} = 114$, targets up to ≈ 0.974 are supported; targets 0.98 and 0.99 require 149 and 299 successes, independent of scorer quality.

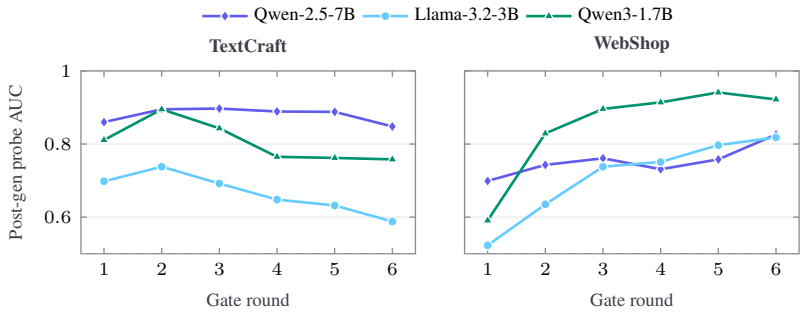

Figure 3: Cross-fitted post-generation probe AUC for predicting eventual episode failure at each gate round in the full 2×3 matrix. The strongest transfer appears on WebShop with Qwen3-1.7B; WebShop Llama becomes informative only after the first two rounds, matching the later-gate savings in Table 1.

We compare against two alternative scorers, holding the entire downstream calibration and search pipeline fixed. The **surface** scorer observes only the agent's *behavior*: a logistic model over trajectory features observable from the serving API alone: the mean action-token log-probability of the current round, the number of generated tokens, the prefix length, and the count of preceding rounds whose environment feedback contains error keywords ("error", "invalid", "fail", etc.). The **stacking** scorer concatenates these surface features onto the probe's activation features, testing whether behavioral evidence adds anything beyond the hidden state. The surface model is deliberately cheap and hand-engineered, matching the negligible-overhead regime our monitor operates in; richer structured-trace monitors are complementary, and an apples-to-apples comparison would place them inside the same recall-calibrated cascade while accounting for their inference overhead.

2.3 Per-Round Recall-Calibrated Gates

Each gate's threshold is set on a calibration split so that the gate provably passes at least a t_r fraction of successful episodes. Given a per-round recall budget t_r, let $\mathcal{S}_r = \{f_r(x_i) : y_i = 1,\ i \text{ alive at } r\}$ be the scores of the n_r successful calibration episodes alive at round r. For a candidate threshold τ, let $k(\tau) = |\{s \in \mathcal{S}_r : s \le \tau\}|$ be the number of survivors; the exact binomial (Clopper–Pearson) lower confidence bound on the gate's true survival rate is the Beta quantile

$$\underline{p}(\tau) \;=\; \mathrm{Beta}^{-1}(\alpha;\ k(\tau),\ n_r - k(\tau) + 1)\,, \tag{1}$$

and we set τ_r to the smallest calibration score with $\underline{p}(\tau_r) \ge t_r$, at per-gate confidence level $1-\alpha$ with $\alpha = 0.05$. This is deliberately stronger than a marginal conformal quantile: the per-round guarantee holds with high confidence over the calibration draw, at the price of conservatism when n_r is small (a gate abstains, i.e., aborts nothing, whenever n_r is insufficient to support t_r). An empirical-quantile variant ($\tau_r = \mathrm{Quantile}_{t_r}(\mathcal{S}_r)$ under the stated failure-score convention) saves more but can violate its target with small calibration sets. The implementation uses the sign-equivalent success-score convention and hence the corresponding lower-tail quantile at $1 - t_r$. We therefore use the high-confidence Clopper–Pearson gate throughout. A budget of $t_r = 1$ disables the gate.

2.4 Recall Budget Search Under a Global Constraint

Per-round guarantees do not compose multiplicatively in any useful way: a union bound over gates remains valid yet is overly conservative, because it ignores that late gates expose few successes and that per-gate false-abort events are far from disjoint. We therefore treat the budget vector $\mathbf{t} = (t_1, \ldots, t_{R_g})$ as a hyperparameter and select it empirically on a validation split disjoint from calibration.

Concretely, probe scores are produced by task-grouped stratified group k-fold cross-fitting, so every episode is scored by a probe that never saw its task during training. Tasks are then partitioned into a calibration set for gate thresholds (20% of tasks), a validation set for budget search (20%), and a held-out test set (60%). Grouping by task throughout ensures no task contributes episodes to two sides of any split.

Environment	Model	Allocation	0.90	0.92	0.95	0.97
TextCraft	Qwen-2.5-7B	Uniform	8.3 ± 4.8	8.3 ± 4.8	0.0 ± 0.0	0.0 ± 0.0
		Single	40.1 ± 7.1	34.8 ± 7.8	25.8 ± 9.0	15.9 ± 10.3
		Cascade	$\mathbf{60.2 \pm 6.8}$	$\mathbf{56.6 \pm 10.1}$	$\mathbf{45.0 \pm 12.5}$	$\mathbf{31.6 \pm 15.0}$
TextCraft	Llama-3.2-3B	Uniform	3.4 ± 2.8	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0
		Single	22.0 ± 6.4	17.1 ± 4.8	10.7 ± 3.8	6.0 ± 3.4
		Cascade	$\mathbf{34.7 \pm 9.2}$	$\mathbf{25.8 \pm 11.0}$	$\mathbf{17.2 \pm 7.2}$	$\mathbf{6.9 \pm 7.1}$
TextCraft	Qwen3-1.7B	Uniform	5.9 ± 6.0	4.6 ± 4.5	0.0 ± 0.0	0.0 ± 0.0
		Single	29.9 ± 8.6	23.2 ± 8.5	14.3 ± 7.9	7.1 ± 5.1
		Cascade	$\mathbf{47.3 \pm 12.2}$	$\mathbf{43.3 \pm 12.9}$	$\mathbf{26.8 \pm 13.0}$	$\mathbf{14.8 \pm 12.7}$
WebShop	Qwen-2.5-7B	Uniform	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0
		Single	19.2 ± 6.9	14.4 ± 5.2	8.0 ± 5.2	5.0 ± 4.2
		Cascade	$\mathbf{35.9 \pm 5.7}$	$\mathbf{31.8 \pm 5.4}$	$\mathbf{20.6 \pm 8.2}$	$\mathbf{11.3 \pm 7.8}$
WebShop	Llama-3.2-3B	Uniform	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0
		Single	2.8 ± 1.8	2.1 ± 1.3	1.1 ± 1.1	0.0 ± 0.0
		Cascade	$\mathbf{24.7 \pm 8.4}$	$\mathbf{20.0 \pm 10.1}$	$\mathbf{13.6 \pm 9.6}$	$\mathbf{12.0 \pm 10.2}$
WebShop	Qwen3-1.7B	Uniform	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0
		Single	8.3 ± 4.8	6.1 ± 4.4	3.8 ± 4.2	1.8 ± 1.3
		Cascade	$\mathbf{54.9 \pm 5.7}$	$\mathbf{52.3 \pm 6.6}$	$\mathbf{41.5 \pm 15.8}$	$\mathbf{26.4 \pm 17.1}$

Table 1: Compute saved (%) for the searched stacking cascade, best single gate, and uniform allocation across all environments and models. The cascade dominates both allocation baselines in all 24 configurations; achieved recall stays within one standard deviation of its target throughout (Figure 5).

For each candidate $\mathbf{t}$ on the grid $t_r \in \{0.85, 0.90, 0.95, 0.98, 0.99, 1.0\}$ ($6^6 = 46{,}656$ candidates), we calibrate all gates on the calibration split, simulate the full cascade on the validation split, and record its global recall $\hat{\rho}_{\text{val}}(\mathbf{t})$ and compute savings. The deployed budget maximizes validation savings subject to a feasibility condition, in a two-stage design that separates flexible search from statistically clean certification:

- **Margin** (default): require $\hat{\rho}_{\text{val}}(\mathbf{t}) \geq \rho^\star + \delta$ with a fixed safety margin $\delta = 0.02$, a guard against selection bias from searching over many candidates. We size the margin from the binomial standard error of validation recall and validate its coverage empirically across the full matrix.
- **Independent post-selection certificate** (optional): after selecting $\mathbf{t}^\star$, freeze the cascade and compute a Clopper–Pearson lower bound at level $1 - \alpha_m$ ($\alpha_m = 0.05$) on its global recall using successful episodes in an independent certification sample; deploy only if the bound exceeds $\rho^\star$. Because the cascade is fixed before these data are observed, the bound is exact and distribution-free regardless of the preceding search size. This converts the margin's empirical control into a formal, a priori verifiable guarantee whenever one is required, without constraining the search itself.

If no candidate is margin-feasible, or if an optional independent certificate fails, the policy abstains and aborts nothing: the method's failure mode is conservative by construction. The full procedure is given as pseudocode in Algorithm 1 (Appendix). Two fixed budget allocations serve as structural baselines: **single-gate**, which spends the entire recall budget at a single round, with both the round and its budget selected by the same validation search (recovering the single-decision-point policies of prior monitoring work as a special case of our framework), and **uniform**, which sets the same t_r at every gate.

Sample complexity of certification. The certificate makes visible a fundamental data requirement. Let n_{pos} be the number of successful episodes in the independent certification sample. Even a candidate that aborts *nothing* has a lower bound on an independent certification sample of $\text{Beta}^{-1}(\alpha_m; n_{\text{pos}}, 1) = \alpha_m^{1/n_{\text{pos}}}$ (the "rule of three"), so targets above this level are unattainable regardless of scorer quality. The resulting sample complexity of stricter targets is analyzed in the Results (Figure 2) and applies to every matrix cell through its own number of certification successes.

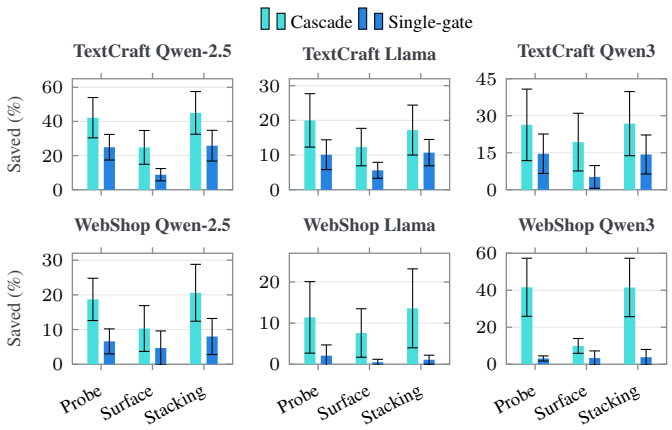

Figure 4: Scorer ablation at target global recall 0.95 across the full 2×3 matrix (mean $\pm$ one standard deviation over 20 seeds). "Stacking" concatenates surface features onto the activation probe.

3 Experimental Setup

Environment and Agents. We evaluate a 2×3 matrix: two environments and three agent policies. TextCraft (Prasad et al., 2024) is a text-based crafting environment from AgentGym (Xi et al., 2024) in which an agent must synthesize a target item by navigating multi-step recipes. WebShop is an instruction-following shopping environment in which the agent searches, clicks products and options, and buys an item matching the user request. The three agent policies are Llama-3.2-3B (Meta, 2024), Qwen-2.5-7B (Qwen Team, 2025), and Qwen3-1.7B. Every cell contains 800 rollout episodes. TextCraft uses 100 tasks with 8 rollouts per task; WebShop uses 200 tasks with 4 rollouts per task, keeping the episode count fixed while increasing task diversity. TextCraft episodes run up to $R_{\text{full}} = 20$ rounds, WebShop episodes run up to 10 rounds, and gates are placed at rounds 1–6 in all cells. The supplementary alive-episode curves show that TextCraft episodes often terminate early, while WebShop episodes remain alive longer. Compute savings are reported as the fraction of total generated tokens saved by aborts, counting only each aborted episode's remaining computation.

Protocol. Probes are per-round logistic regressions with $C = 1$ and standardized features, applied to the fixed layer selected for each model family: layer 14 for Llama-3.2-3B, layer 20 for Qwen-2.5-7B, and layer 28 for Qwen3-1.7B in the full-matrix runs. On one server with eight NVIDIA H20 GPUs, the entire pipeline (task-level splitting, probe cross-fitting, gate calibration, budget search, evaluation) is repeated over 20 random seeds; we report test-split mean $\pm$ standard deviation. All scorers and all budget-allocation baselines share identical splits, calibration machinery, and search procedure, so differences isolate the quality of the underlying signal (scorer comparisons) or the value of distributing the recall budget (allocation comparisons). Unless otherwise stated, the allocation tables use the stacking scorer (activation probe plus surface features) as the common scorer: it contains both feature families, giving behavioral evidence every opportunity to contribute. Figure 4 shows the probe alone performs equivalently, so the reported gains attribute to the activations. For Qwen3-1.7B we use non-thinking generation mode for action-format compatibility; for WebShop with Llama-3.2-3B we use a format-constrained WebShop instruction to reduce invalid actions.

4 Results

4.1 Internal States Predict Failure Before Behavior Does

Figure 3 plots the cross-fitted post-generation probe AUC for predicting eventual failure among episodes alive at each round, for every cell. In every cell the probe becomes strongly informative within the first rounds, with timing that varies across cells; this heterogeneity is itself informative. TextCraft Qwen-2.5-7B and Qwen3-1.7B are already highly separable in the first two rounds (AUC 0.86 and 0.81 at round 1), while WebShop Llama-3.2-3B becomes informative only after round 2.

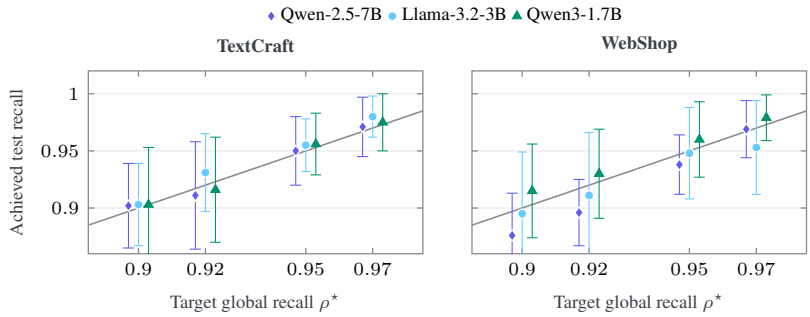

Figure 5: Achieved global success recall of the searched cascade versus its target in the full 2×3 matrix (mean $\pm$ one standard deviation over 20 seeds; points are horizontally jittered for legibility). The diagonal marks exact targeting; every configuration lands within one standard deviation of its target.

WebShop Qwen3-1.7B is the strongest transfer case: its AUC rises from 0.591 at round 1 to 0.896 at round 3 and above 0.92 at later gates. These differences explain why the savings frontier in Table 1 differs across cells, and they are precisely why a *searched* allocation should beat any fixed decision point: early gates protect more remaining compute, but some cells require one or two rounds before the hidden-state signal is reliable, and the budget search discovers this per cell without manual tuning. For WebShop Llama-3.2-3B, the format-constrained instruction reduces invalid actions. The present experiment jointly reflects format compliance and task comprehension; its near-chance early AUC establishes the limited outcome predictiveness of the monitored state at this stage, while leaving their individual contributions unresolved.

4.2 Main Results: Cascade vs. Single Gate

Table 1 compares the searched cascade with two baselines, the best single gate and uniform allocation, over the complete 2×3 matrix. For comparability, all use the same activation-based stacking scorer, Clopper–Pearson gate, grouped splits, 20 seeds, and margin $\delta = 0.02$; Section 4.3 separately isolates the value of the activation signal. Figure 5 plots achieved recall. Three observations stand out.

First, *the cascade dominates both allocation baselines in every cell and at every target*: 24 of 24 configurations. At the 90% target its advantage over the best single gate ranges from $1.5\times$ (TextCraft Qwen-2.5-7B, 60.2% vs. 40.1%) to $8.8\times$ (WebShop Llama-3.2-3B, 24.7% vs. 2.8%). The gap is largest exactly where the signal matures late (Figure 3): a single gate must commit to one round, whereas the cascade defers budget to the rounds where each cell becomes predictable. The uniform allocation, forced to be equally strict at recall-expensive early gates and low-yield late ones, collapses to near-zero savings, confirming that savings depend critically on both the amount and placement of the recall budget.

Second, *achieved recall tracks its target throughout* (Figure 5): across all 24 configurations the cascade's mean test recall deviates from its target by at most 0.024, always within one seed-level standard deviation, and errs on the conservative side in TextCraft. The largest deviations occur on WebShop Qwen-2.5-7B at loose targets, where validation successes are scarcest; they are well within seed noise, and the independent post-selection certificate of Section 4.5 exists precisely to convert this empirical tracking into a formal guarantee whenever one is required. The savings in Table 1 are thus attained under honest, pre-specified risk control.

Third, *savings follow signal quality more closely than success rate*. The strongest cells (TextCraft Qwen-2.5-7B at 60.2%, WebShop Qwen3-1.7B at 54.9%) are those whose probes are most separable in the rounds that guard the most remaining compute; weaker cells retain smaller but still positive savings while preserving their recall floor.

4.3 Does the Signal Require Internal Access?

Figure 4 fixes the target at 0.95 and swaps only the scorer inside the otherwise-identical pipeline. Two conclusions emerge. First, *behavior-only monitoring is consistently weaker than an activation probe*, most sharply on WebShop Qwen3-1.7B (41.6% vs. 9.9% saved, a $4.2\times$ gap) and TextCraft Qwen-2.5-7B (42.2% vs. 24.8%). Second, *stacking surface features onto the probe matches it and yields no further improvement* in every cell, indicating that whatever the surface scorer knows, the hidden states already encode; the converse is clearly false. For the behavioral features tested here, internal activations therefore retain additional predictive information. A diagnostic at the strictest target finds that the lower-dimensional probe additionally transfers more reliably from validation to test when recall headroom is small, so we recommend probe-only scoring when targets are strict. This ablation compares against specified low-cost features; structured prefix monitors remain an interesting extension within the same cascade.

4.4 Stress Test: Failing Safe in a Low-Success Regime

A reliable recall-controlled abort policy must also preserve successes when the signal is weak. We use ALFWorld (Shridhar et al., 2021) as a deliberate boundary test of this property. On 477 Qwen-2.5-7B rollouts from 120 tasks, the early internal signal recurs in a third, harder environment (probe AUC 0.815), and at the nominal 0.95 target the cascade saves 10.7% versus 3.3% for a single gate, with mean recall 0.943. For the weaker Qwen3-1.7B (AUC 0.764), successful calibration examples are scarce, and the cascade contracts to 1–4% savings. This is the designed behavior: when the data cannot support aggressive aborting, the recall-calibrated gates abstain and the policy approaches a no-op, prioritizing successes. These experiments are recalibrated in-domain; zero-shot transfer remains untested.

4.5 The Cost of Independent Post-Selection Certification

After margin-based search, a frozen cascade can be certified on an independent sample ($\alpha_m = 0.05$), and the certificate makes the data cost of strict promises explicit. Figure 2 plots the required number of successful episodes, $n_{\text{pos}} \geq \ln \alpha_m / \ln \rho^\star$. With 114 successes, targets up to ≈ 0.974 are certifiable; 0.98 and 0.99 require roughly 149 and 299, independent of scorer quality. When the sample is insufficient, the valid choices are to abstain, collect more data, or relax the target. This transparency makes the data requirement operational: the same machinery that saves compute also reports, before deployment, which recall promises the available data support and prescribes exactly how much additional data a stricter promise costs.

5 Related Work

Internal representations. Activation probes recover latent knowledge, truth, hallucination, and outcome signals (Kadavath et al., 2022; Burns et al., 2023; Marks & Tegmark, 2024; Azaria & Mitchell, 2023; Orgad et al., 2025; Afzal et al., 2025; Zhang et al., 2025). Agent representations also expose success/failure directions (Padhi et al., 2026); we turn this early signal into a stopping rule with controlled retained-success recall.

Agent monitoring. Agent monitors use text auditors, weak supervision, runtime statistics, prefix warnings, and structured traces (Zhang et al., 2026; Baidya et al., 2026; Li et al., 2026; Barke et al., 2026; Chen et al., 2026; Pham et al., 2026; Mittapalli et al., 2026; Dellibarda Varela et al., 2026; Huang et al., 2026; Zhao, 2026). Our monitor reads frozen-policy activations without an auxiliary LLM pass and jointly calibrates recall over the *entire sequence* of abort decisions.

Risk control. Distribution-free calibration provides finite-sample guarantees under exchangeability (Angelopoulos & Bates, 2023; Bates et al., 2021; Angelopoulos et al., 2024; 2025). CALM controls sequence-level token exits, FIPER calibrates rollout alarms, and dynamic abstention learns when reasoning should quit (Schuster et al., 2022; Römer et al., 2025; Davidov et al., 2026). Our setting controls episode-level retained-success recall across agent gates and independently certifies the selected cascade.

Adaptive compute. Adaptive inference methods stop self-consistency, route between models, predict ongoing success, prune partial trajectories, or end futile reasoning (Aggarwal et al., 2023; Li et al., 2024; Manvi et al., 2024; Chen et al., 2023a; Zellinger et al., 2025; Kim & Yoo, 2026; Lightman et al., 2024; Guan et al., 2026; Lu et al., 2025; Shrivastava, 2026; Lin et al., 2026; Sun et al., 2025; Mao et al., 2025). We decide whether a running agent episode is worth finishing and pair saved tokens with calibrated success recall.

6 Discussion and Limitations

Statistical control and selection. The default margin provides empirical control, validated in all 24 matrix configurations. An independent post-selection sample upgrades a frozen cascade to a formal distribution-free guarantee. Formal validity comes from the independent certificate, while reused search data support empirical selection. When successes are scarce, grouped splits can select different discrete budgets; more successful tasks, repeated grouped selection, and certification respectively reduce selection variance and control the final policy's recall.

Our coarse grid fails conservatively: a suboptimal feasible point preserves recall at the expense of savings. Finer search may improve a stepwise frontier, although exhaustive cost grows as $|\mathcal{T}|^{R_g}$; coordinate, greedy, or beam search are natural alternatives. We fixed $\delta = 0.02$ a priori, near the binomial standard error at our scale ($\rho^\star = 0.95$, $n_{\text{pos}} \approx 114$). Under distribution shift, the probe, thresholds, and global control must all be refreshed on labeled data.

Scope and system realization. Our balanced evidence spans two environments, three models, and 800 episodes per cell, with ALFWorld as a boundary test. Grouped splits avoid task leakage but remain in-distribution; category-held-out tasks are a natural next test. Distribution shift may also require reselecting the pilot-fixed layer, and multi-layer features or later gates for longer trajectories would require recalibration.

Teacher-forced replay supports our offline evaluation. For online use, vLLM provides native selected-layer hidden-state extraction during inference (Schmitt-Ulms, 2026); our monitor consumes one final-token vector per turn. Systems cost remains nonzero because the documented extraction path persists activations to shared storage and currently requires chunked prefill to be disabled. In the offline replay experiments, our efficiency metric is therefore generated-token savings; wall-clock latency and dollar cost are not estimated from replay.

What aborted compute buys. The present evaluation measures saved compute. Reallocating it to retries would create a test-time-scaling policy whose reward and systems effects require end-to-end measurement. We additionally include a preliminary asynchronous RL-training case study showing that the cascade can improve reward while reducing average rollout tokens and per-round wall-clock time.

7 Conclusion

This work addresses early stopping for LLM agents while controlling the loss of episodes that would eventually succeed. We introduced a recall-controlled cascade that combines hidden-state failure probes, jointly allocated per-round recall budgets, and independent post-selection certification. Across TextCraft and WebShop with three model families, the probes predicted eventual failure earlier than behavior-only monitors. The cascade outperformed the best single gate in all 24 configurations and reduced generated tokens by up to 60.2% at 90% global recall, while remaining conservative when evidence was weak. These results show that internal-state monitoring can support controllable adaptive inference. Future work should test distribution shift, measure end-to-end serving savings, and evaluate whether recovered compute can improve task success through retries or additional sampling.

AI USE STATEMENT

Generative AI was used to aid and polish the writing of this manuscript, and for retrieval and discovery of related work. All technical content, experimental results, and conclusions are the authors' own, and the authors are responsible for the final content.

REPRODUCIBILITY STATEMENT

The appendix specifies training settings, sample counts, and scoring conventions. The analysis supplement supplies observed outcomes and scripts for the figures and tables.

REFERENCES

Anum Afzal, Florian Matthes, Gal Chechik, and Yftah Ziser. Knowing before saying: LLM representations encode information about chain-of-thought success before completion. In *Findings of the Association for Computational Linguistics: ACL*, 2025.

Pranjal Aggarwal, Aman Madaan, Yiming Yang, and Mausam. Let's sample step by step: Adaptive-consistency for efficient reasoning and coding with LLMs. In *Empirical Methods in Natural Language Processing*, 2023.

Guillaume Alain and Yoshua Bengio. Understanding intermediate layers using linear classifier probes. *arXiv preprint arXiv:1610.01644*, 2016.

Anastasios N Angelopoulos and Stephen Bates. Conformal prediction: A gentle introduction. *Foundations and Trends in Machine Learning*, 16(4):494–591, 2023.

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

Anastasios N Angelopoulos, Stephen Bates, Emmanuel J Candès, Michael I Jordan, and Lihua Lei. Learn then test: Calibrating predictive algorithms to achieve risk control. *The Annals of Applied Statistics*, 19(2):1641–1662, 2025.

Amos Azaria and Tom Mitchell. The internal state of an LLM knows when it's lying. In *Findings of the Association for Computational Linguistics: EMNLP*, 2023.

Avinash Baidya, Xinran Liang, Ruocheng Guo, Xiang Gao, and Kamalika Das. When evidence is sparse: Weakly supervised early failure alerting in dialogs and LLM-agent trajectories. *arXiv preprint arXiv:2606.05414*, 2026.

Shraddha Barke, Arnav Goyal, Alind Khare, Avaljot Singh, Suman Nath, and Chetan Bansal. AgentRx: Diagnosing AI agent failures from execution trajectories. *arXiv preprint arXiv:2602.02475*, 2026.

Stephen Bates, Anastasios Angelopoulos, Lihua Lei, Jitendra Malik, and Michael I Jordan. Distribution-free, risk-controlling prediction sets. *Journal of the ACM*, 68(6):1–34, 2021.

Yonatan Belinkov. Probing classifiers: Promises, shortcomings, and advances. *Computational Linguistics*, 48(1):207–219, 2022.

Collin Burns, Haotian Ye, Dan Klein, and Jacob Steinhardt. Discovering latent knowledge in language models without supervision. In *International Conference on Learning Representations*, 2023.

Lingjiao Chen, Matei Zaharia, and James Zou. FrugalGPT: How to use large language models while reducing cost and improving performance. *arXiv preprint arXiv:2305.05176*, 2023a.

Shuguang Chen et al. Signals: Trajectory sampling and triage for agentic interactions. *arXiv preprint arXiv:2604.00356*, 2026.

Yanxi Chen et al. EE-LLM: Large-scale training and inference of early-exit large language models with 3D parallelism. *arXiv preprint arXiv:2312.04916*, 2023b.

John J Cherian, Isaac Gibbs, and Emmanuel J Candès. Large language model validity via enhanced conformal prediction methods. In *Advances in Neural Information Processing Systems*, 2024.

Hen Davidov, Nachshon Cohen, Oren Kalinsky, Yaron Fairstein, Guy Kushilevitz, Ram Yazdi, and Patrick Rebeschini. Knowing when to quit: A principled framework for dynamic abstention in LLM reasoning. *arXiv preprint arXiv:2604.18419*, 2026.

DeepSeek-AI. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning. *Nature*, 645:633–638, 2025.

Iñaki Dellibarda Varela, R. Sendra-Arranz, Pablo Romero-Sorozabal, J. M. Valverde-García, Annemarie F. Laudanski, Álvaro Gutiérrez, Eduardo Rocon, and Manuel Cebrian. POIROT: Interrogating agents for failure detection in multi-agent systems. *arXiv preprint arXiv:2606.02282*, 2026. URL https://arxiv.org/abs/2606.02282.

Xinyan Guan, Jiali Zeng, Chunlei Xin, Yaojie Lu, Hongyu Lin, Xianpei Han, Le Sun, and Fandong Meng. Knowing when to quit: Diagnosing and training LLMs to abort futile reasoning. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 16823–16835, 2026. doi: 10.18653/v1/2026.findings-acl.830. URL https://aclanthology.org/2026.findings-acl.830/.

Xinmiao Huang, Jinwei Hu, Rajarshi Roy, Changshun Wu, Yi Dong, and Xiaowei Huang. PrefixGuard: From LLM-agent traces to online failure-warning monitors. *arXiv preprint arXiv:2605.06455*, 2026. URL https://arxiv.org/abs/2605.06455.

Ziwei Ji, Delong Chen, Etsuko Ishii, Samuel Cahyawijaya, Yejin Bang, Bryan Wilie, and Pascale Fung. LLM internal states reveal hallucination risk faced with a query. In *Proceedings of the 7th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*, 2024.

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

Naryeong Kim and Shin Yoo. Atropos: Improving cost-benefit trade-off of LLM-based agents under self-consistency with early termination and model hotswap. *arXiv preprint arXiv:2604.15075*, 2026.

Jannik Kossen, Jiatong Han, Muhammed Razzak, Lisa Schut, Shreshth Malik, and Yarin Gal. Semantic entropy probes: Robust and cheap hallucination detection in LLMs. *arXiv preprint arXiv:2406.15927*, 2024.

Jing Lei, Max G'Sell, Alessandro Rinaldo, Ryan J Tibshirani, and Larry Wasserman. Distribution-free predictive inference for regression. *Journal of the American Statistical Association*, 113 (523):1094–1111, 2018.

Xianyou Li, Weiran Yan, Yichao Wu, Penghao Liang, Mengwei Yuan, Jianan Liu, and Jing Yang. Early diagnosis of wasted computation in multi-agent LLM systems via failure-aware observability. *arXiv preprint arXiv:2606.01365*, 2026.

Yiwei Li, Peiwen Yuan, Shaoxiong Feng, Boyuan Pan, Xinglin Wang, Bin Sun, Heda Wang, and Kan Li. Escape sky-high cost: Early-stopping self-consistency for multi-step reasoning. In *International Conference on Learning Representations*, 2024.

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

Yuxiang Lin, Zihan Wang, Mengyang Liu, Yuxuan Shan, Longju Bai, Junyao Zhang, Xing Jin, Boshan Chen, Jinyan Su, Xingyao Wang, Jiaxin Pei, and Manling Li. BAGEN: Are LLM agents budget-aware? *arXiv preprint arXiv:2606.00198*, 2026. URL https://arxiv.org/abs/2606.00198.

Qingyu Lu, Liang Ding, Siyi Cao, Xuebo Liu, Kanjian Zhang, Jinxia Zhang, and Dacheng Tao. Runaway is ashamed, but helpful: On the early-exit behavior of large language model-based agents in embodied environments. In *Findings of the Association for Computational Linguistics: EMNLP*, 2025.

Enze Ma, Yufan Zhou, Wei-Chieh Huang, Jie Yang, Huanhuan Ma, Zixuan Wang, Chengze Li, Chunyu Miao, Philip S. Yu, and Zhen Wang. MEMPROBE: Probing long-term agent memory via hidden user-state recovery. *arXiv preprint arXiv:2606.24595*, 2026.

Rohin Manvi, Anikait Singh, and Stefano Ermon. Adaptive inference-time compute: LLMs can predict if they can do better, even mid-generation. *arXiv preprint arXiv:2410.02725*, 2024.

Minjia Mao, Bowen Yin, Yu Zhu, and Xiao Fang. Early stopping chain-of-thoughts in large language models. *arXiv preprint arXiv:2509.14004*, 2025.

Samuel Marks and Max Tegmark. The geometry of truth: Emergent linear structure in large language model representations of true/false datasets. In *Conference on Language Modeling*, 2024.

Aman Mehta. When agents commit too soon: Diagnosing premature commitment in LLM agents. *arXiv preprint arXiv:2606.22936*, 2026.

Meta. Llama-3.2-3B model card. https://huggingface.co/meta-llama/Llama-3.2-3B, September 2024.

Ruijie Miao, Yihan Yan, Xinshuo Yao, and Tong Yang. An efficient inference framework for early-exit large language models. *arXiv preprint arXiv:2407.20272*, 2024.

Vijitha Mittapalli, Shreyaa Jayant Dani, Satya Srujana Pilli, Snigdha Ansu, Mohammadreza Teymoorianfard, Franck Dernoncourt, Hongjie Chen, Yu Wang, Ryan A. Rossi, and Nesreen K. Ahmed. TRACE: Trajectory reasoning through adaptive cross-step evidence aggregation for LLM agents. *arXiv preprint arXiv:2606.07054*, 2026. URL https://arxiv.org/abs/2606.07054.

Christopher Mohri and Tatsunori Hashimoto. Language models with conformal factuality guarantees. In *International Conference on Machine Learning*, 2024.

Hadas Orgad, Michael Toker, Zorik Gekhman, Roi Reichart, Idan Szpektor, Hadas Kotek, and Yonatan Belinkov. LLMs know more than they show: On the intrinsic representation of LLM hallucinations. In *International Conference on Learning Representations*, 2025.

Trilok Padhi, Ramneet Kaur, Krishiv Agarwal, Adam D. Cobb, Daniel Elenius, Manoj Acharya, Colin Samplawski, Alexander M. Berenbeim, Nathaniel D. Bastian, Susmit Jha, and Anirban Roy. From actions to understanding: Conformal interpretability of temporal concepts in LLM agents. *arXiv preprint arXiv:2604.19775*, 2026.

Harris Papadopoulos, Kostas Proedrou, Volodya Vovk, and Alex Gammerman. Inductive confidence machines for regression. In *European Conference on Machine Learning*, 2002.

Dzung Pham, Kleomenis Katevas, Ali Shahin Shamsabadi, and Hamed Haddadi. AgentStop: Terminating local AI agents early to save energy in consumer devices. *arXiv preprint arXiv:2605.15206*, 2026. URL https://arxiv.org/abs/2605.15206.

Archiki Prasad, Alexander Koller, Mareike Hartmann, Peter Clark, Ashish Sabharwal, Mohit Bansal, and Tushar Khot. ADaPT: As-needed decomposition and planning with language models. In *Findings of the Association for Computational Linguistics: NAACL 2024*, 2024.

Yuxiao Qu, Tianjun Zhang, Naman Garg, and Aviral Kumar. Recursive introspection: Teaching language model agents how to self-improve. *arXiv preprint arXiv:2407.18219*, 2024. URL https://arxiv.org/abs/2407.18219.

Victor Quach, Adam Fisch, Tal Schuster, Adam Yala, Jae Ho Sohn, Tommi S Jaakkola, and Regina Barzilay. Conformal language modeling. In *International Conference on Learning Representations*, 2024.

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.

Ralf Römer, Adrian Kobras, Luca Worbis, and Angela P. Schoellig. Failure prediction at runtime for generative robot policies. In *Advances in Neural Information Processing Systems*, 2025. URL https://arxiv.org/abs/2510.09459.

Fynn Schmitt-Ulms. Extracting hidden states from vLLM. vLLM Blog, Mar 2026. URL https://vllm.ai/blog/2026-03-30-extract-hidden-states. Accessed: 2026-07-29.

Tal Schuster, Adam Fisch, Jai Gupta, Mostafa Dehghani, Dara Bahri, Vinh Q Tran, Yi Tay, and Donald Metzler. Confident adaptive language modeling. In *Advances in Neural Information Processing Systems*, 2022.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Xu Shen, Qi Zhang, Song Wang, Zhen Tan, Xinyu Zhao, Laura Yao, Vaishnav Tadiparthi, Hossein Nourkhiz Mahjoub, Ehsan Moradi Pari, Kwonjoon Lee, and Tianlong Chen. Metacognitive self-correction for multi-agent system via prototype-guided next-execution reconstruction. In *Findings of the Association for Computational Linguistics: ACL 2026*, 2026.

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. URL https://openreview.net/forum?id=0IOX0YcCdTn.

Sahil Shrivastava. Semantic early-stopping for iterative LLM agent loops. *arXiv preprint arXiv:2606.27009*, 2026.

Renliang Sun, Wei Cheng, Dawei Li, Haifeng Chen, and Wei Wang. Stop when enough: Adaptive early-stopping for chain-of-thought reasoning. *arXiv preprint arXiv:2510.10103*, 2025.

Vladimir Vovk, Alexander Gammerman, and Glenn Shafer. *Algorithmic Learning in a Random World*. Springer, 2005.

Zhiheng Xi, Yiwen Ding, Wenxiang Chen, Boyang Hong, Honglin Guo, Junzhe Wang, Dingwen Yang, Chenyang Liao, Xin Guo, Wei He, Songyang Gao, Lu Chen, Rui Zheng, Yicheng Zou, Tao Gui, Qi Zhang, Xipeng Qiu, Xuanjing Huang, Zuxuan Wu, and Yu-Gang Jiang. AgentGym: Evolving large language model-based agents across diverse environments. *arXiv preprint arXiv:2406.04151*, 2024.

Yu Xia, Jingru Fan, Weize Chen, Siyu Yan, Xin Cong, Zhong Zhang, Yaxi Lu, Yankai Lin, Zhiyuan Liu, and Maosong Sun. AgentRM: Enhancing agent generalization with reward modeling. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2025.

Michael J. Zellinger, Rex Liu, and Matt Thomson. Cost-saving LLM cascades with early abstention. *arXiv preprint arXiv:2502.09054*, 2025.

Anqi Zhang, Yulin Chen, Jane Pan, Chen Zhao, Aurojit Panda, Jinyang Li, and He He. Reasoning models know when they're right: Probing hidden states for self-verification. *arXiv preprint arXiv:2504.05419*, 2025.

Boxuan Zhang, Jianing Zhu, Zeru Shi, Dongfang Liu, and Ruixiang Tang. Agentforesight: Online auditing for early failure prediction in multi-agent systems. *arXiv preprint arXiv:2605.08715*, 2026.

Dan Zhang, Sining Zhoubian, Ziniu Hu, Yisong Yue, Yuxiao Dong, and Jie Tang. ReST-MCTS*: LLM self-training via process reward guided tree search. In *Advances in Neural Information Processing Systems*, 2024.

Yue Zhao. GRADE: Graph representation of LLM agent dependency and execution. *arXiv preprint arXiv:2606.22741*, 2026. URL https://arxiv.org/abs/2606.22741.

Kunlun Zhu, Zijia Liu, Bingxuan Li, Muxin Tian, Yingxuan Yang, Jiaxun Zhang, Pengrui Han, Qipeng Xie, Fuyang Cui, Weijia Zhang, Xiaoteng Ma, Xiaodong Yu, Gowtham Ramesh, Jialian Wu, Zicheng Liu, Pan Lu, James Zou, and Jiaxuan You. Where LLM agents fail and how they can learn from failures. *arXiv preprint arXiv:2509.25370*, 2025.

PSEUDOCODE

Algorithm 1 Recall-Controlled Abort Cascade with Optional Certification

Input: labeled episodes $\mathcal{D}$, global recall target $\rho^\star$, budget grid $\mathcal{T}$, margin δ; optional independent certification set $\mathcal{D}_{\mathrm{cert}}$ and level α_m

Output: gates $\{(f_r, \tau_r)\}_{r=1}^{R_g}$

```
Score all episodes by task-grouped cross-fitted probes f_r
Partition tasks into calibration / search-validation / test; keep D_cert independent if supplied
for each candidate budget t ∈ T do
    for r = 1, ..., R_g do
        τ_r(t) ← smallest threshold whose Clopper–Pearson lower bound on per-round survival of
        successful calibration episodes at r is ≥ t_r
    end for
    Simulate cascade on validation split; record global recall ρ̂_val and savings
end for
F ← candidates with ρ̂_val ≥ ρ* + δ
if F = ∅ then
    Abstain (no aborts)
else
    Deploy t* = arg max_{t∈F} validation savings
end if
if D_cert is supplied and its global Clopper–Pearson lower bound for t* is < ρ* then
    Abstain (no aborts)
end if
Evaluate once on the test split
```

A. PER-LAYER PROBE AUC SWEEPS

Probe layers were fixed once per model and then used unchanged in all recall-target and cascade evaluations. For Qwen-2.5-7B, a per-layer sweep over layers $\{0, 2, \ldots, 28\}$ was run on an independent pilot set collected before the main experiments, selecting layer 20. For Llama-3.2-3B, the sweep covered layers $\{6, 10, 14, 18, 22, 26\}$ and selected layer 14. For Qwen3-1.7B, the full-matrix run swept layers $\{4, 8, \ldots, 28\}$ and selected layer 28.

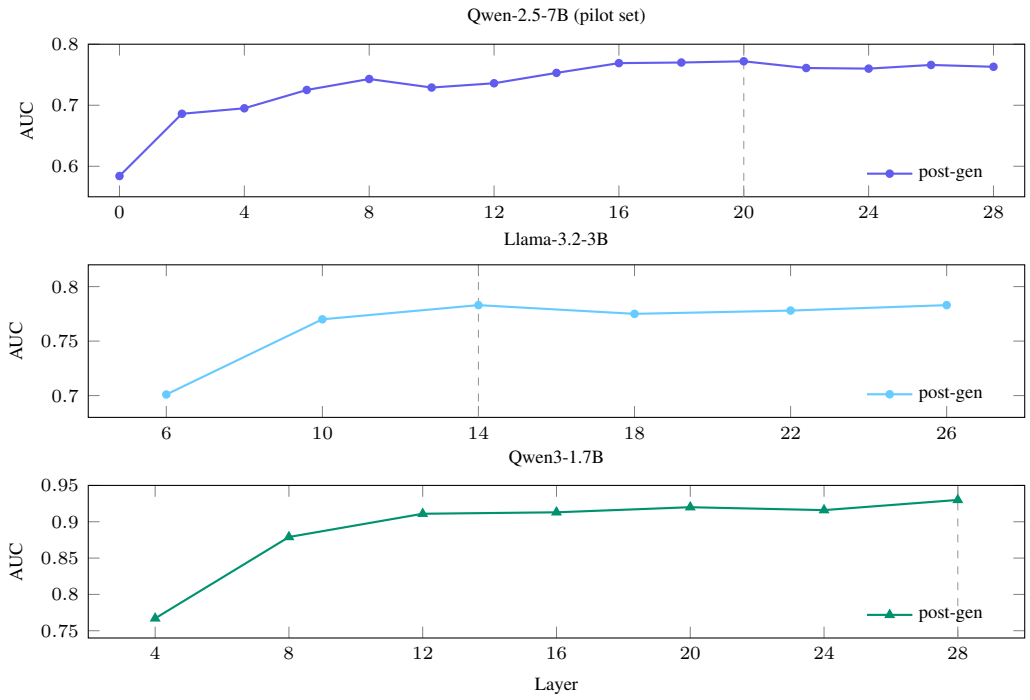

Figure 1: Per-layer probe-AUC sweeps used to fix the probe layer for each agent. Qwen-2.5-7B: independent pilot set, layer 20. Llama-3.2-3B: layer 14. Qwen3-1.7B: layer 28.

B. Empirical-Quantile Gates

Table 1 compares the Clopper–Pearson gate of the main text against the empirical-quantile variant, for a single probe-based gate at round 1 with per-round target 0.95 (20 seeds). The quantile gate controls recall only on average over the calibration draw, so with the small per-round calibration sets available here its realized recall fluctuates below target, violating it for Llama (0.933 vs. 0.95) and Qwen3-1.7B (0.940 vs. 0.95) while saving more compute. The Clopper–Pearson gate is conservative (recall 0.966–0.977 at target 0.95) but never violates in these round-1 comparisons. This is the trade the main text accepts: the cascade's budget search recovers much of the conservatism by spending recall where it is cheap, while retaining a high-confidence per-gate guarantee.

Model	Calibration	Recall	Saved (%)
Llama-3.2-3B	Clopper–Pearson	0.972 ± 0.013	10.4 ± 3.9
	Quantile	0.933 ± 0.029	20.7 ± 5.6
Qwen-2.5-7B	Clopper–Pearson	0.977 ± 0.011	17.4 ± 5.5
	Quantile	0.954 ± 0.020	24.7 ± 4.8
Qwen3-1.7B	Clopper–Pearson	0.966 ± 0.020	14.6 ± 8.0
	Quantile	0.940 ± 0.032	27.7 ± 9.8

Table 1: Single gate at round 1, per-round recall target 0.95, probe scorer, 20 seeds. Quantile calibration saves more but violates the target on Llama-3.2-3B and Qwen3-1.7B; Clopper–Pearson is conservative and meets it in all three model families.

C. Scorer Robustness at Strict Recall Targets

At $\rho^\star = 0.95$, stacking surface features onto the probe neither helps nor hurts. At the strictest target the comparison changes character: with $\rho^\star = 0.97$ the headroom between the target and perfect recall is only 0.03, shifting the focus to whether the budget selected on validation still controls recall

on test. Table 2 compares three scorers on TextCraft with Qwen-2.5-7B at $\rho^\star = 0.97$: a logistic activation probe, an MLP variant of the probe (identical features with a nonlinear head: a single hidden layer of 256 units on the same standardized features), and the stacking scorer.

In mean savings the three scorers are statistically indistinguishable at this target (17.2%, 16.7%, and 15.5%, each with a standard deviation above ten points; the largest paired difference, MLP vs. stacking, is 1.7 points with standard error 2.6, $t(19) = 0.67$), so the comparison rests on per-seed recall control. There the scorers order consistently, though the differences are not individually significant at 20 seeds: the MLP probe meets the floor in all 20 seeds (minimum test recall 0.971), the logistic probe in 17 of 20 (minimum 0.960), and stacking in 16 of 20 (minimum 0.957). Because all three scorers are evaluated on the same 20 splits, the appropriate test is paired: in the extreme comparison (MLP, 0 violations, vs. stacking, 4), all four discordant seeds favor the MLP, and an exact one-sided sign test gives $p = 0.062$, directional but not significant at this sample size. Two of the violating seeds are shared by the logistic probe and stacking, so split difficulty rather than scorer identity drives most violations. (This batch shares its seed stream with the margin sweep in Appendix D; under the shift identity, the logistic row here coincides seed-for-seed with that sweep's $\delta = 0.02$ entry at this target, 3 of 20 in Table 3. The seed-for-seed agreement provides a deterministic consistency check on the implementation.) What distinguishes stacking is the depth of its worst failure: its worst seed selects a budget whose validation recall is 1.000 yet achieves only 0.957 on test, a transfer gap of 0.043 that exceeds the entire 0.03 headroom between the target and perfect recall, against a worst logistic gap of 0.031 and no violation at all for the MLP probe. Every violating seed of either scorer selected a budget with validation recall at least 0.991, so these failures are invisible at search time; they are precisely the selection optimism the margin is sized to absorb in the margin-size sweep of Appendix D. The higher-dimensional stacked features give the $46{,}656$-candidate budget search more room to overfit the validation split, and since the surface features add no discriminative signal beyond the activations, the extra dimensions buy variance without information. The practical prescription is simple: when recall headroom is small, prefer the lower-dimensional probe (or its MLP variant), which delivers the same savings with tighter recall control.

This robustness batch uses the seed stream shared with the margin-size sweep and serves as a separate diagnostic of strict-target behavior.

Scorer	Recall	Saved (%)	Min recall	Seeds $\geq \rho^\star$
MLP probe	0.989 ± 0.010	17.2 ± 10.8	0.971	$20/20$
Logistic probe	0.985 ± 0.011	16.7 ± 10.5	0.960	$17/20$
Stacking	0.986 ± 0.014	15.5 ± 14.0	0.957	$16/20$

Table 2: Cascade at $\rho^\star = 0.97$ on TextCraft with Qwen-2.5-7B (20 seeds, test split; margin $\delta = 0.02$ as in the main experiments). "Seeds $\geq \rho^\star$" counts seeds whose test recall meets the floor.

D. Margin Size Sweep

For fixed splits and seeds, the margin rule's feasible set $\{\mathbf{t} : \hat{\rho}_{\text{val}}(\mathbf{t}) \geq \rho^\star + \delta\}$ depends on $(\rho^\star, \delta)$ only through their sum, so the budget deployed under margin δ at target $\rho^\star$ is identical, seed by seed, to the unmargined budget at target $\rho^\star + \delta$. A single unmargined sweep over a fine target grid therefore characterizes *every* margin size at once: choosing δ means choosing where on the curves of Figure 2 to operate, and the cost of a margin is the savings drop between $\rho^\star$ and $\rho^\star + \delta$.

Table 3 reads off the per-seed violation rates. Unmargined selection violates the floor in up to $12/20$ seeds; $\delta = 0.01$ helps but still admits up to $8/20$; $\delta = 0.02$ caps violations at $4/20$ at every target, and at $3/20$ for $\rho^\star \geq 0.91$. Its price follows the slope of the savings curve: near-zero at loose targets (about one point of savings at $\rho^\star = 0.90$) and steepest where the frontier falls fastest (about ten points at $\rho^\star = 0.95$ in this batch). This sweep uses its own fixed seed stream to characterize margin sensitivity; the main-body allocation numbers come from a separate evaluation.

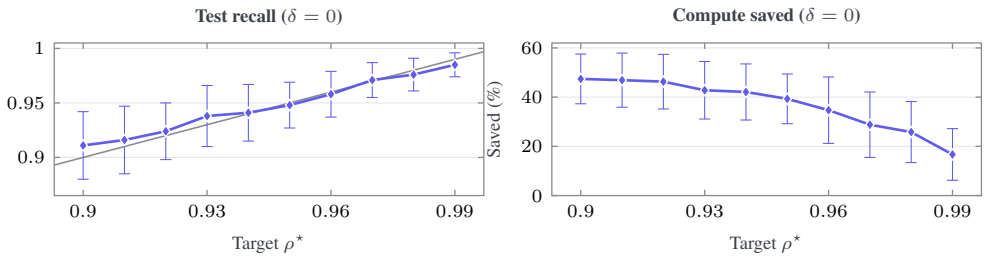

Figure 2: Unmargined ($\delta = 0$) cascade on TextCraft with Qwen-2.5-7B over a fine target grid (mean $\pm$ one standard deviation, 20 seeds). Left: test recall hugs the diagonal, dipping below it at several targets; the margin primarily absorbs per-seed dispersion around the mean (Table 3), while mean-level bias is negligible. By the shift identity, deploying margin δ at target $\rho^\star$ reads both panels at $\rho^\star + \delta$.

	Seeds below floor (of 20)		
$\rho^\star$	$\delta = 0$	$\delta = 0.01$	$\delta = 0.02$
0.90	8	7	4
0.91	8	6	3
0.92	7	4	2
0.93	6	4	1
0.94	8	4	3
0.95	10	7	1
0.96	12	5	3
0.97	10	8	3

Table 3: Number of seeds (of 20) whose test recall falls below the target, per margin size; all columns are derived from the single unmargined sweep via the shift identity. $\delta = 0.02$ caps violations at $4/20$ everywhere, and at $3/20$ for $\rho^\star \geq 0.91$.

E. ALFWorld Stress-Test Details

Table 4 gives the full per-target results behind the paper's ALFWorld stress test. The table reports the Qwen models evaluated with the recorded multi-rollout protocol. The separate Llama pilot is discussed only as a calibration boundary because its success population is too small for a meaningful recall-calibrated cascade evaluation. The fixed subset contains the first 20 tasks from each of six high-level task families (120 tasks total), with reset failures retained in the attempted-rollout counts. In the Llama pilot, a format-focused retry on a more favorable solved-any subset solves only $10/68$ tasks (14.7%), reinforcing its role as an out-of-scope, low-success calibration boundary.

Model	Target	Test recall	Cascade	Single	Success/attempts
Qwen-2.5-7B	0.90	0.891 ± 0.047	$\mathbf{19.7 \pm 5.3}$	11.4	$250/477$
	0.92	0.900 ± 0.052	$\mathbf{18.3 \pm 6.8}$	7.9	
	0.95	0.943 ± 0.045	$\mathbf{10.7 \pm 6.9}$	3.3	
	0.97	0.969 ± 0.046	$\mathbf{5.5 \pm 6.5}$	1.1	
Qwen3-1.7B	0.90	0.960 ± 0.076	$\mathbf{4.3 \pm 8.1}$	3.7	$106/477$
	0.92	0.960 ± 0.076	$\mathbf{4.3 \pm 8.1}$	2.0	
	0.95	0.974 ± 0.062	$\mathbf{2.4 \pm 4.9}$	0.0	
	0.97	0.974 ± 0.062	$\mathbf{2.4 \pm 4.9}$	0.0	

Table 4: ALFWorld stress test (compute saved in %). Both models use 477 recorded rollouts; success/attempts reports the base success population.

F. Probe Data Cost and Break-Even Accounting

The pipeline requires labeled complete trajectories before deployment. The environment already supplies the binary success label, and the same logged trajectories are reused for cross-fitting, calibration, and budget search. In our offline implementation, however, hidden activations are recovered by a teacher-forced replay, so the one-time cost includes both collecting the original trajectories and replaying them for feature extraction. Fitting the linear probes is inexpensive relative to LLM generation, but trajectory collection can still be costly, especially for an independent certification set.

Let C_{data} denote the total one-time cost of trajectory collection, activation extraction, and probe fitting in a common unit such as generated-token-equivalents. Let $\bar{C}$ be the mean baseline cost per deployed episode, s the expected saved fraction, and C_{mon} the per-episode monitoring overhead in the same unit. The deployment breaks even after

$$N_{\text{BE}} = \frac{C_{\text{data}}}{s\bar{C} - C_{\text{mon}}}, \qquad s\bar{C} > C_{\text{mon}}.$$

If the denominator is non-positive, the monitor never repays its data and systems cost. Existing trajectory logs reduce C_{data} to incremental extraction and fitting. Certification is separate: n_{pos} successes require roughly $n_{\text{pos}}/p_{\text{succ}}$ attempts at base success rate p_{succ}. Because serving overhead is unmeasured, token savings and end-to-end cost should be interpreted separately.

G. Additional Main-Text Diagnostics

The following plots preserve two complete diagnostics omitted from the seven-page technical narrative for space. Figure 3 shows how much future compute remains at each candidate gate, while Figure 4 expands the main paper's numerical allocation comparison across every recall target.

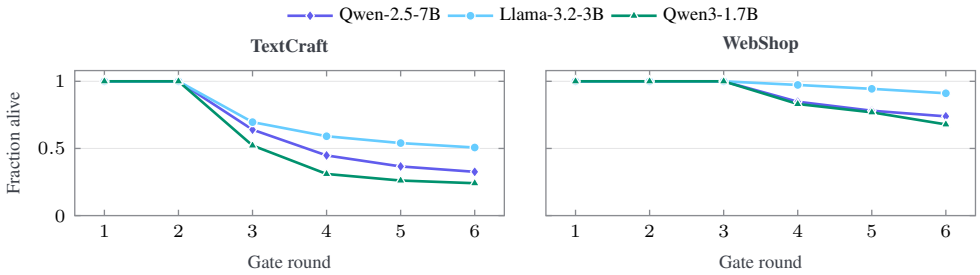

Figure 3: Fraction of episodes still running at each gate round in the full 2×3 matrix. TextCraft episodes often terminate early, especially for Qwen3-1.7B, while WebShop episodes remain alive longer; this controls how much compute a later gate can save.

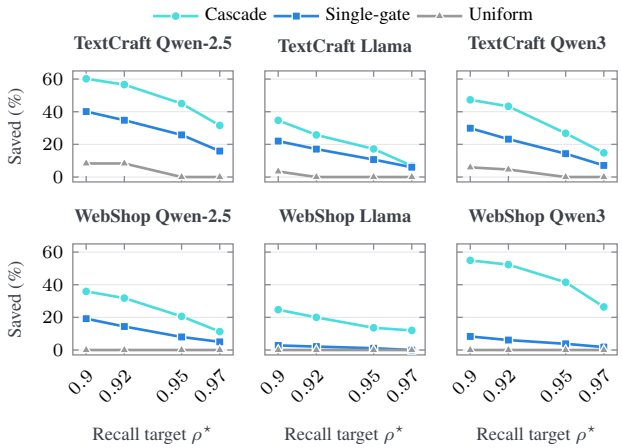

Figure 4: Compute savings versus global recall target for all three allocations and matrix cells. The cascade dominates single-gate and uniform allocation in every configuration.

H. Additional Related Work

Predicting success from internal representations. Frozen-activation probes decode knowledge, truth, and hallucination signals that models do not reliably verbalize (Alain & Bengio, 2016; Belinkov, 2022; Kadavath et al., 2022; Azaria & Mitchell, 2023; Burns et al., 2023; Marks & Tegmark, 2024; Orgad et al., 2025; Kossen et al., 2024; Ji et al., 2024). Internal states also anticipate chain-of-thought success and encode self-verification (Afzal et al., 2025; Zhang et al., 2025). In agent settings, Padhi et al. (2026) recover success/failure directions for representation steering, while MEMPROBE targets persistent memory rather than online residual activations (Ma et al., 2026), and Mehta (2026) study commitment, deliberately separated from correctness. These results establish that outcome-relevant signal exists in the residual stream and is linearly accessible; we ask when it becomes actionable and turn it into a stopping rule with controlled error.

Failure detection and monitoring of LLM agents. Decisive agent errors often arise early and compound (Zhu et al., 2025; Li et al., 2026); AgentRx localizes such steps post hoc (Barke et al., 2026). Online approaches range from text or embedding auditors, including AgentForesight, weakly supervised alerters, and MASC (Zhang et al., 2026; Baidya et al., 2026; Shen et al., 2026), to behavioral statistics and low-cost runtime signals (Chen et al., 2026; Pham et al., 2026). Richer monitors use diagnostic agents, typed events, or execution graphs (Mittapalli et al., 2026; Dellibarda Varela et al., 2026; Huang et al., 2026; Zhao, 2026); other methods alter refusal, stop on repetition, or elicit self-reports (Guan et al., 2026; Shrivastava, 2026; Lin et al., 2026; Lu et al., 2025). Our activation reader needs no auxiliary LLM pass, and its cascade controls retained-success recall over all checks.

Conformal prediction and risk control for LLMs. Split conformal and related calibration give finite-sample guarantees under exchangeability (Vovk et al., 2005; Papadopoulos et al., 2002; Lei et al., 2018; Angelopoulos & Bates, 2023), with extensions to general risks (Bates et al., 2021; Angelopoulos et al., 2025; 2024). LLM applications include prediction sets, factuality filters, and help-seeking planners (Quach et al., 2024; Mohri & Hashimoto, 2024; Cherian et al., 2024; Ren et al., 2023). Davidov et al. (2026) derive optimal quitting from a learned value function. CALM composes token exits under a sequence-level constraint (Schuster et al., 2022), while FIPER calibrates robot-policy alarms from successful rollouts (Römer et al., 2025). Our fixed-policy certificate instead controls retained-success recall across an episode-level sequence of gates.

Adaptive allocation of inference compute. RLVR and recovery training optimize sampled trajectories from task outcomes (Shao et al., 2024; DeepSeek-AI, 2025; Qu et al., 2024). At inference time, adaptive self-consistency stops once answers stabilize (Aggarwal et al., 2023; Li et al., 2024; Manvi et al., 2024), model cascades route by difficulty (Chen et al., 2023a; Zellinger et al., 2025), and overthinking methods stop redundant reasoning (Sun et al., 2025; Mao et al., 2025). Atropos pre-

dicts ongoing reasoning success (Kim & Yoo, 2026); early-exit systems vary decoder depth (Chen et al., 2023b; Miao et al., 2024); and process reward models prune partial trajectories (Lightman et al., 2024; Zhang et al., 2024; Xia et al., 2025). We instead decide whether one running agent episode is worth finishing and pair savings with calibrated retained-success recall.

I. Asynchronous Reinforcement-Learning Training Case Study

Beyond offline replay and fixed benchmark episodes, we also evaluate the cascade during asynchronous reinforcement-learning training. The run uses the veRL framework to train Qwen2.5-3B-Instruct on ALFWorld with GRPO, using 16 groups, 8 rollouts per group, and learning rate 10^{-6}. We use veRL's vLLM backend because, as noted in the main text, this backend supports direct hidden-state extraction. The method is particularly useful in asynchronous RL: low-promise trajectories can be stopped early, and the trainer can begin a model update without waiting for every rollout in the round to finish. This deployment-oriented study applies the same calibrated early-abort logic inside the rollout loop, where aborted trajectories reduce the token budget consumed by each training round and shorten the iteration cycle. Figure 5 aligns the recorded online traces by cumulative wall-clock time. Over the wall-clock interval covered by both runs, validation success improves while mean sequence length and average rounds decrease relative to the reference trace.

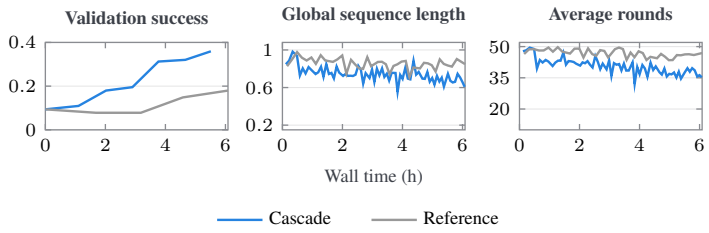

Figure 5: Asynchronous reinforcement-learning training case study over the shared wall-clock interval. All panels use cumulative wall-clock time on the x-axis. Curves show the logged validation-success, global-sequence-length, and average-round values; validation curves share the same initial success point.