When Fixed Thresholds Fail: Trace-Grounded Scheduling for Multi-Tenant LLM Fine-Tuning under SLOs

Anonymous authors
Paper under double-blind review

Abstract

Multi-tenant platforms can schedule LLM fine-tuning by watching each job's evaluation curve, terminating runs whose score declines and handing the freed GPUs to another tenant. A recently published rule of this kind reports a 1.5% false-positive rate. We replay it on 78 real per-checkpoint trajectories from public pretraining runs. Every run is healthy, so every termination is an error, and the rule fires on 44.9% of them. Neither obvious explanation holds up. The synthetic concave curve used for calibration predicts an even higher rate, and real evaluation noise is milder than assumed rather than harsher. What actually carries the published number is a quantity it never states, the per-interval signal-to-noise ratio. Real curves sit at a median of 0.55, whereas 1.5% at $k = 2$ demands 1.17. Correct that single parameter and the original generator reproduces our measurement to within four points. Raising k does not rescue the rule, because real declines cluster, and a classical quickest-detection bound puts the delay needed for a 1.5% false-alarm rate at three times the checkpoints a job has, so no causal detector reaches the published operating point on these curves. Standard reporting would not have caught this. Replaying a $714{,}903$-job production trace, we find that how SLO attainment is aggregated moves the reported number further than which scheduler produced it, by as much as 17.7 points, and reorders the policies outright. Early termination then corrupts the usual metrics in one direction. On a workload with no declining jobs at all, the published rule "improves" mean job completion time by 16.7% while destroying 44% of useful compute. At the trace's actual prevalence of decline, nothing we tested pays for itself, including a learned policy that never beats a one-parameter rule normalising drawdown by an online noise estimate. We recommend net goodput reported with a false-positive rate quoted at its checkpoint cadence, the one pairing that survived every probe we ran.

1 Introduction

Multi-tenant platforms now host many concurrent LLM fine-tuning jobs, and a scheduler that can read each job's evaluation curve has an appealing lever. A run whose eval score has started to degrade is consuming GPUs another tenant could use, so terminating it should improve everything at once. Recent work makes exactly this argument, wrapping any base scheduler in a rule that terminates a job after k consecutive declines and reporting a 1.5% false-positive rate alongside substantial latency gains (Zhang et al., 2026).

That result is established in simulation, against a synthetic model of a healthy training run: a monotone concave mean plus i.i.d. Gaussian noise. The paper's own limitations section names replaying the detector on real training traces as the primary missing validation. We supply a replay of that kind, and the rule does not survive it. Our 78 per-checkpoint trajectories come from a public model suite (Biderman et al., 2023), all ordinary pretraining runs with no reward model, so every termination is by construction a false positive. At its default $k = 2$ the rule terminates $\mathbf{44.9}\%$ of them.

Neither candidate we expected explains the gap. The synthetic concave shape predicts a higher rate than we observe, and real evaluation noise is milder than the calibration assumed. What carries the

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.

published number is a quantity it never states, the measured per-interval signal-to-noise ratio, and at the regime real curves occupy a classical quickest-detection bound puts the published operating point out of reach for any causal detector. Learning the policy instead does not help, because the limit lies in the observation process and not the policy class.

Testing the rule on a real workload exposed a second problem, this one in the objective rather than the detector. Replaying the Alibaba PAI 2020 production trace (Weng et al., 2022) of $714{,}903$ jobs from $1{,}219$ tenants, we find that reported SLO attainment depends more on *how it is aggregated* than on which scheduler produced it: holding the schedules fixed and changing only the aggregation moves attainment by up to 17.7 percentage points and reorders the policies outright. Early termination then corrupts these metrics in one direction. On a workload with *no* declining jobs, where every termination is pure damage, the published rule "improves" mean job completion time by 16.7% while destroying 44% of the cluster's useful work.

This paper contributes that refutation and its attribution (Section 4), the measurement of objective fragility (Sections 3 and 6), a break-even accounting and the detectability floor behind the negative learning result (Section 6), and an open trace-driven simulator with a mechanically enforced information boundary (Section 5).

2 Background and Related Work

Cluster scheduling for deep learning. DL-cluster schedulers have converged on a few signals: remaining-time estimates, elasticity, and heterogeneity. Tiresias (Gu et al., 2019) drops the need for accurate duration estimates, Gavel (Narayanan et al., 2020) treats heterogeneity as a policy input, Pollux (Qiao et al., 2021) and Sia (Jayaram Subramanya et al., 2023) optimize goodput, and Shockwave (Zheng et al., 2023) targets fairness under adaptation, all within the non-clairvoyant framing (Motwani et al., 1994) we enforce mechanically (Section 5). Learned schedulers do well on structured workloads (Mao et al., 2019) and algorithms-with-predictions formalises what a prediction must be worth (Purohit et al., 2018), but Im et al. (2021) show low prediction error need not translate into scheduling benefit, the mechanism we suspect behind our own null result and one with precedent in cloud resource prediction (Christofidi et al., 2023). The Philly (Jeon et al., 2019), Alibaba PAI (Weng et al., 2022) (which we use) and Acme (Hu et al., 2024b) traces anchor the literature empirically, with Blox (Agarwal et al., 2024) the closest thing to a sim-versus-real fidelity standard. We take the base scheduler as given and study the *termination* decision and the *aggregation* of the SLO objective.

Curve-based early termination. Terminating a run whose curve looks unpromising is well studied in multi-fidelity hyperparameter optimisation, from successive halving (Jamieson & Talwalkar, 2016) and Hyperband (Li et al., 2018) to ASHA (Li et al., 2020), with Vizier's median stopping rule (Golovin et al., 2017) a deployed fixed-threshold heuristic, predictive termination extrapolating the curve instead (Domhan et al., 2015; Klein et al., 2017), and Hydro (Hu et al., 2023) bringing this machinery to LLM-scale tuning. All of them rank a run *against concurrent siblings* from a known search space, whereas a multi-tenant scheduler must judge one tenant's run in isolation, and none faces a declining true-quality signal underneath a monotone proxy. Our negative result concerns that isolated-run regime.

Quality-aware scheduling and eval-triggered stopping. That a scheduler should read training quality rather than resource counters alone goes back at least to SLAQ (Zhang et al., 2017), and classical early stopping (Prechelt, 1998) halts one run on a validation criterion, whereas the scheduler-level analogue reallocates the freed capacity. The immediate predecessor applies that analogue to multi-tenant RLHF, terminating a job after k consecutive eval-score declines (Zhang et al., 2026). Its limitations section anticipates two of our findings qualitatively, expecting autocorrelated eval noise to inflate its false-positive rate and confidence-aware thresholds to widen its operating range, and it names replaying the detector on real RLHF/DPO traces as the primary missing validation. We claim the quantification, not the discovery: we measure how large both effects are on real curves and identify the assumed signal-to-noise regime, rather than the curve shape or noise magnitude, as what carries the reported operating point.

Sequential change detection. Deciding online whether a noisy sequence has changed is quickest detection (Page, 1954), whose theory ties detection delay to the false-alarm rate and to the per-observation information in the change (Lorden, 1971; Tartakovsky et al., 2014), while anytime-valid confidence sequences supply thresholds under a stated error budget (Howard et al., 2021). Our baseline (Eq. 3) is a CUSUM in this family, and Section 4 uses the delay bound to explain why no fixed threshold reaches the published operating point.

Reward overoptimization and multi-tenant platforms. The phenomenon these detectors target is well documented, as optimizing against a learned proxy reward eventually degrades true quality (Gao et al., 2023; Skalse et al., 2022) in both RLHF (Ouyang et al., 2022) and direct alignment algorithms (Rafailov et al., 2023; 2024), motivating constrained variants (Moskovitz et al., 2024). We take degrading runs as given and study only scheduling-time detection. Meanwhile parameter-efficient methods (Hu et al., 2022) and multi-adapter serving (Sheng et al., 2024) make it economical for one cluster to host many tenant fine-tuning jobs, and open RLHF stacks (Hu et al., 2024a) make the eval-in-the-loop pattern standard. Our trajectories come from published per-checkpoint results (Biderman et al., 2023) produced with a standard harness (Gao et al., 2021).

3 Problem Formulation

Job and schedule model. A workload is a set of jobs $\mathcal{J}$; job j is a tuple $(a_j, u_j, g_j, S_j, \mathcal{C}_j, c_j)$ of arrival time, tenant, GPU demand (fractional), service time at full allocation, eval-checkpoint sequence, and SLO class. A schedule π assigns start times $s_j^\pi \geq a_j$, giving waiting time $w_j^\pi = s_j^\pi - a_j$, completion $C_j^\pi = s_j^\pi + S_j$, and $\mathrm{JCT}_j^\pi = w_j^\pi + S_j$. We adopt a non-preemptive, no-slowdown execution model, as the predecessor's simulator does, so S_j is schedule-invariant. Offered load over a window of length T on G GPU-equivalents is $\rho = \frac{1}{GT}\sum_j g_j S_j$.

Per-class SLO and its aggregations. Each class c carries a wait budget W_c; job j's deadline is $d_j = a_j + W_{c_j} + S_j$ and j is attained iff $C_j^\pi \leq d_j$, giving the indicator χ_j^π. Attainment admits several inequivalent aggregations:

$$A^\pi_{\text{job}} = \tfrac{1}{N}\textstyle\sum_j \chi_j^\pi, \quad A^\pi_{\text{ten}} = \tfrac{1}{|\mathcal{U}|}\sum_u \tfrac{1}{|\mathcal{J}_u|}\sum_{j\in\mathcal{J}_u}\chi_j^\pi,$$
$$A^\pi_{\text{gpu}} = \frac{\sum_j g_j S_j \chi_j^\pi}{\sum_j g_j S_j}, \quad Q^\pi_q = \mathrm{Quantile}_q\big(\{A_u^\pi\}_u\big). \tag{1}$$

These are the micro-average, the macro-average a per-tenant SLA implies, a value-weighted average, and a tail-tenant measure. That the choice matters in principle is not new, being the substance of "nines are not enough" (Mogul & Wilkes, 2019), of reporting rules on when means summarise costs versus rates (Hoefler & Belli, 2015), and of composition effects that reorder ML leaderboards (Dehghani et al., 2021). What we add is its size: Section 6 shows these aggregations reorder policies by more than the schedulers differ, which makes the choice a modelling commitment rather than a presentational detail.

Useful checkpoints and wasted compute. Given a quality threshold θ, the first useful checkpoint is $k_j^\star(\theta) = \min\{k : y_{j,k} \geq \theta\}$ and $\mathrm{TTFUC}_j^\pi(\theta) = w_j^\pi + \tau_{j,k_j^\star(\theta)}$; wasted compute is GPU-time after a job's last useful checkpoint, $\mathrm{Waste}^\pi = \sum_j g_j (E_j^\pi - \tau_{j,\ell_j^\pi})_+$. TTFUC is the predecessor's construct, adopted unchanged, with one caveat we establish empirically: it must be measured *relative to submission*, since in absolute terms every policy reads 134–140ks, a quantity that is $\approx 95\%$ arrival time, while relative to submission the same policies separate by $1.9\times$.

Detection as a constraint. A scheduler may terminate a job at a checkpoint. With T^π the terminated set and $\mathcal{H}$ the unobserved healthy set, $\mathrm{FPR}^\pi = |T^\pi \cap \mathcal{H}|/|\mathcal{H}|$. We treat detection as a *constraint* rather than a second objective, since falsely killing a healthy job destroys tenant value no latency gain compensates. The scheduler optimizes

$$\min_\pi \ \mathcal{L}(\pi) = \sum_c \omega_c (1 - A_c^\pi) \ + \ \lambda \widetilde{\mathrm{Waste}}^\pi \ + \ \mu \sum_c \omega_c \frac{\mathrm{TTFUC}_c^\pi(\theta)}{W_c} \qquad \text{s.t.} \quad \mathrm{FPR}^\pi \leq \varepsilon, \tag{2}$$

with class weights ω_c, every term computable online from observable quantities. Setting $\lambda = \mu = 0$ and $\omega_c \propto |\mathcal{J}_c|$ recovers per-job attainment; $\omega_c = 1/3$ gives equal class weight.

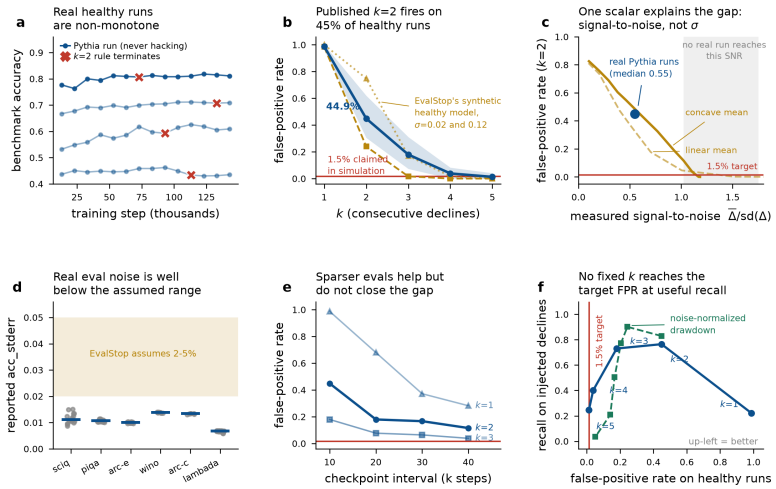


Figure 1: **A fixed decline threshold does not survive contact with real evaluation curves.** (a) Real pretraining runs, healthy by construction, are non-monotone; red crosses mark where the published $k = 2$ rule terminates them. (b) That rule fires on 44.9% of 78 known-healthy trajectories against the 1.5% reported in simulation (red line). (c) One scalar, the measured per-interval signal-to-noise ratio $\overline{\Delta}/\mathrm{sd}(\Delta)$, explains the gap, because the published operating point requires a regime no real run reaches (grey). (d) Reported per-checkpoint standard errors are *below* the 2–5% the calibration assumed, so the failure is not noise magnitude. (e) Sparser evaluation reduces but does not close the gap. (f) No fixed k reaches the target false-positive rate at useful recall.

Relation to the predecessor's position. The predecessor defines TTFUC and argues that JCT/TTFUC-only framings miss the detection axis. We agree, and our claim is additive: adopting a detection axis is necessary but not sufficient, because *how* the per-job quantities are aggregated into one scalar is itself a modelling decision, and here the standard choices are dominated by the bulk class and a few large tenants. One hypothesis of ours the data refuted. We expected mean JCT to *misrank* policies against attainment, and it does not (Kendall's $\tau = 0.82$); it is *insensitive* rather than wrong, since with S_j schedule-invariant only $\overline{w}^{\pi}$ is controllable and the invariant service term is 96.6% of mean JCT at $\rho = 0.9$ (Appendix D).

4 Real Evaluation Trajectories Break Fixed-Threshold Stopping

4.1 Setup: Real Curves That Are Healthy by Construction

The predecessor's stop rule was calibrated against a synthetic healthy run, a monotone concave mean plus i.i.d. Gaussian noise, with σ swept over 0.01–0.12 on the stated grounds that typical LLM benchmark standard errors are 2–5%. Its own limitations section names replaying the detector on real RLHF/DPO traces as the primary missing validation. We supply that replay in the only regime for which public per-checkpoint trajectories exist, pretraining rather than fine-tuning (Section 4.5).

We use published per-checkpoint results for the Pythia suite (Biderman et al., 2023), evaluated with a standard harness (Gao et al., 2021). These are ordinary pretraining runs with no reward model, so *every* termination is a false positive, an unusually clean negative set. We admit 6 of the 65 available tasks, dropping those that never leave a band around chance, and we evaluate a uniform 14-checkpoint tail, since Pythia's early checkpoints are logarithmically spaced whereas a scheduler observes evenly spaced ones. Both choices are conservative and leave 78 model $\times$ task trajectories (Appendix A).

4.2 The Published Rule Fires on Nearly Half of Healthy Runs

Replaying the rule exactly as published gives, at its default $k = 2$, a false-positive rate of $\mathbf{44.9}\%$ (Wilson 95% CI $[34.3, 55.9]$; 35 of 78) against the 1.5% reported in simulation. The two rates share

a denominator, both being the fraction of *healthy jobs* terminated. At $k = 3$, the DPO default, 17.9% of trajectories still trip. Our trajectories are 6 tasks $\times$ 13 model scales rather than 78 independent draws, so the interval is optimistic about precision, but the effect is an order of magnitude larger than any plausible clustering correction.

Noise magnitude is not the problem. Measured per-checkpoint standard errors are 0.58%–1.51% (Figure 1d), *below* the 2–5% the calibration assumed, and a rule calibrated for noisier observations than it receives should fire *less*, not more.

4.3 Attribution: the signal-to-noise regime, not the curve shape

We first tested the natural hypothesis, that the synthetic monotone concave *shape* was at fault. It is not. Re-implementing the predecessor's generator at its own stated parameters gives 55.0% at $\sigma = 0.05$, *above* the real 44.9%. Set honestly, it over-predicts false positives.

What does explain the gap is the measured per-interval signal-to-noise ratio $\mathrm{SNR} = \overline{\Delta}/\mathrm{sd}(\Delta)$, where Δ are successive score changes. Real trajectories have median measured $\mathrm{SNR} = 0.55$ and never exceed 1.02. Sweeping the predecessor's own generator over measured SNR (Figure 1c), $\mathrm{SNR} = 0.55$ predicts 48.7%, within 4 points of what we observe. The noise *model* is adequate; the assumed *regime* is not. Reaching 1.5% at $k = 2$ requires measured $\mathrm{SNR} \geq 1.17$ under a concave mean, which no real run approaches.

A second, independent failure is that real declines **cluster**. Calibrating an i.i.d. generator to reproduce the real $k = 2$ rate exactly, runs of three declines still occur $3.4\times$ more often in real curves and runs of four $14.5\times$ more often. Raising k, the obvious fix, therefore buys less than the i.i.d. model predicts.

4.4 No fixed threshold has an acceptable operating point

Measuring recall against synthetically injected declines grafted onto the same real trajectories, the fixed-k family traces a poor frontier, from 98.7% FPR at 22% recall for $k = 1$, through 44.9% at 76% for $k = 2$, to 1.3% at 25% for $k = 5$ (Figure 1f). A one-parameter alternative dominates the whole family. Normalise the drawdown from a job's running best by an online estimate of the noise scale, and terminate at checkpoint k when

$$\frac{\max_{k' \leq k} y_{k'} \; - \; y_k}{\widehat{\mathrm{sd}}_k(\Delta)} \; > \; c, \qquad \widehat{\mathrm{sd}}_k(\Delta) = \mathrm{sd}\big(\{\Delta_{k'}\}_{k' \leq k}\big), \tag{3}$$

where c is the single parameter and the noise scale is estimated causally from the job's own prefix, so the rule reads nothing a deployed scheduler could not. At $c = 1.5$ it reaches 24.4% FPR at 90% recall, still far short of 1.5%, which shows that adapting to the noise scale helps where a constant cannot. Any fixed-k FPR is also **undefined without stating the checkpoint budget**, since $k = 2$ moves from 3.8% at $n = 4$ checkpoints to 44.9% at $n = 14$ (Figure 1e), so operating points reported without a cadence are not reproducible.

That every rule falls short has a classical explanation. Decline detection is a quickest-detection problem (Page, 1954), and Lorden's minimax bound (Lorden, 1971) makes the optimal detector's expected delay scale as $2 \log \gamma / \mathrm{SNR}^2$ against an average run length γ to false alarm. At measured $\mathrm{SNR} = 0.55$, holding 1.5% over 14 checkpoints demands a delay of order 45 checkpoints, over three times what such a job has, so the published operating point is not simply miscalibrated: no causal detector attains it. The bound rests on two assumptions, one of which our own clustering result violates, so we read it as an order-of-magnitude statement rather than a theorem (Appendix B).

4.5 What transfers, and what does not

Pythia trajectories are *pretraining* rather than RLHF fine-tuning, the largest limitation of this diagnosis. What transfers is the statistical claim that a fixed-k false-positive rate is a function of the measured SNR regime, the dependence structure of the deltas, and the checkpoint budget, none of which the published calibration accounts for. A reader who believes RLHF curves sit at measured $\mathrm{SNR} = 1.0$ reads 13.4% off Figure 1c instead of 44.9%, still $9\times$ the published 1.5%.

5 EXPERIMENTAL SETUP

Workload. We replay the Alibaba PAI 2020 GPU-cluster trace (Weng et al., 2022). After retaining only jobs with an uncensored duration and a positive GPU request, the workload is 714,903 jobs from 1,219 tenants over 68.4 days (4.73M GPU-hours). We do not trim the tail, because the longest 1% of jobs hold 35.7% of GPU-seconds and trimming would remove the scheduling problem itself. Three properties contradict the synthetic setup the predecessor calibrated against, each adverse for the rules under test. Arrivals are not Poisson but batch-driven and long-range dependent (Hurst exponent 0.75). Durations have decreasing hazard (Weibull shape $c = 0.48 < 1$), so elapsed runtime *positively* predicts remaining time, which undercuts the remaining-time heuristics these stop rules compose with. And demand is concentrated and fractional, with Gini 0.930 on job counts and 37.4% of jobs requesting less than one GPU (Appendix C).

SLO classes. The trace carries no SLO field, so we derive classes rather than assume them. A 1-D kernel density estimate on log duration is trimodal, giving *interactive* (21.5% of jobs, 0.06% of GPU-hours), *batch* (46.2%, 3.89%) and *bulk* (32.3%, 96.05%), with labels $\geq$ 97.8% stable across the threshold window. Each class receives a wait budget W_c equal to its median service time (45 / 452 / 7,354 s), making $d_j = a_j + W_{c_j} + S_j$ a 2× bounded slowdown at the class median. Class is assigned per job, since the median tenant has only 70% of its jobs in its dominant class.

Simulator. A discrete-event simulator replays real arrivals, never a generated arrival process, on a device-slot cluster that packs fractional requests explicitly, a job taking $\lfloor g \rfloor$ whole devices plus a best-fit remainder on one further device. We report the resulting fragmentation cost (0.43pp of peak achievable utilisation) rather than assuming it away, and calibrate capacity from the trace's own occupancy. The main configuration is a three-day window of 33,384 jobs. Two checks bound fidelity: with capacity removed all 714,903 jobs complete at exactly $a_j + S_j$, and windows must be warm-started, since an arrival-only cut understates offered load by 11.8% (Appendix E).

Eval trajectories. Jobs are assigned real published per-checkpoint trajectories (Biderman et al., 2023; Gao et al., 2021), resampled onto each job's checkpoint budget. Because checkpoints are emitted every 600 s of service time, only 39.8% of jobs reach the four checkpoints a decline rule needs, and coverage is stratified by class: 99.0% of bulk jobs are eval-evaluable against 2.0% of batch jobs and no interactive job. Every early-stopping result here is therefore a statement about the bulk class, which is also 96.05% of GPU-hours. Declining trajectories are injected synthetically, since no public corpus of ground-truth reward-hacking runs exists, so recall characterises the shape of the FPR/recall trade-off rather than absolute detector performance.

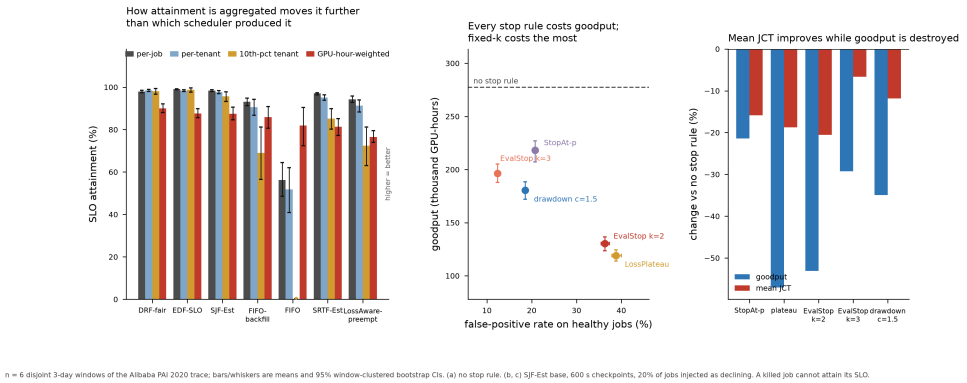
Information boundary. Policies see a frozen view exposing only what a deployed scheduler could observe: arrival, tenant, GPU request, elapsed wait and service, SLO class, and eval scores at checkpoints already reached. True service time, hacking labels, future scores and total checkpoint count raise an error on access, and every policy, the learned one included, is enrolled in a test suite that fails if any is touched.

Metrics. We report goodput (GPU-hours spent on jobs that ran to completion) as the primary systems metric, because mean job completion time is corrupted under early termination: on a workload with *zero* declining jobs the published fixed-k rule still "improves" mean JCT while destroying over 40% of goodput (§6.3). Alongside goodput we report SLO attainment under all four aggregations of Section 3, wasted compute, waiting time and bounded slowdown, tenant fairness, and detection precision/recall/FPR.

6 RESULTS

6.1 AGGREGATION MOVES THE ANSWER FURTHER THAN THE SCHEDULER DOES

Table 1 reports seven schedulers with no stop rule, and the spread across any row exceeds the spread down any column. The per-job minus GPU-hour-weighted gap is significant for all seven, and *its sign depends on the policy*: +17.74pp for LossAware-preempt but −25.65pp for FIFO, which



n = 6 disjoint 3-day windows of the Alibaba PAI 2020 trace; bars/whiskers are means and 95% window-clustered bootstrap CIs. (a) no stop rule. (b, c) SJF-Est base, 600 s checkpoints, 20% of jobs injected as declining. A killed job cannot attain its SLO.

Figure 2: **Main results over six disjoint three-day windows of the trace** (means with 95% window-clustered bootstrap CIs). (a) With no stop rule, the choice of aggregation moves reported SLO attainment further than the choice of scheduler does, and reorders the policies. (b) Every stop rule costs net goodput; the published fixed-k default costs the most among rules with useful recall. (c) On a workload with *no* declining jobs, where every termination is pure damage, mean JCT improves while goodput is destroyed.

starves many small jobs while the few enormous ones still finish inside their generous bulk-class budgets. The reordering is not cosmetic. EDF-SLO leads per job (99.0%) while DRF-fair leads by GPU-hours (90.1%), and further down the ranks cross outright, with FIFO last per job yet fifth of seven by GPU-hours, *above* SRTF-Est. A reader shown one aggregation would choose a different scheduler. The gap also grows with load, from 10.88pp at $\rho = 0.95$ to 56.86pp at $\rho = 2.0$, where SJF-Est's 89.9% per-job attainment delivers only 33.1% of GPU-hours on time.

6.2 Every stop rule costs goodput, and the published default costs most

All seven schedulers deliver *identical* net goodput, 277,468.6 GPU-hours to every printed digit, in every window. This is a property of the execution model, not a measurement: where no job is dropped and service demand is exogenous, a scheduler changes *when* work happens and never *whether*. Gross goodput lacks that property, since it credits work re-run after a preemption, an accounting error we made first ourselves. Net goodput therefore discriminates stop rules, not schedulers.

Table 1 (right) composes each stop rule over the same SJF-Est base at a 600 s cadence with 20% of evaluable jobs injected as declining. Every rule loses net goodput in every window and seed, all CIs excluding zero. The best of them (StopAt-p, which barely detects anything) still costs 21.4%, and the published $k = 2$ default costs 53.1%. No rule breaks even at any injected declining fraction in $[0, 0.4]$. A stop rule buys compute on its true positives and destroys it on false positives, so it pays for itself only above a break-even prevalence each rule sets for itself, and at this trace's prevalence ($\pi = 0.198$) *every* rule we tested sits below its own break-even (Figure 5d). The cause is not bad tuning: at the measured SNR the false positives needed to catch a decline cost more than the decline does.

The one-parameter noise-normalized drawdown rule dominates the published default on *both* axes, with false-positive rate 0.185 against 0.363 and recall 0.840 against 0.789, for 38.5% more net goodput. Our own data forces a qualifier, since EvalStop at $k = 3$ beats the drawdown rule on goodput (+15,858 GPU-h) and on false-positive rate (−6.14pp) while giving up 19.1pp of recall. The defensible claim is therefore *not* that adaptive rules beat fixed thresholds in general, but that they beat the published default, and that the choice of k matters more than the choice of family.

6.3 Four standard metrics rank the most damaging rule near the top

Setting the declining fraction to zero makes every termination pure damage, so any metric that improves is exposing its own defect. This probe is a separate configuration from Table 1, with a

Table 1: Left: schedulers with no stop rule; net goodput is identical across all seven to every printed digit (§6.2). Right: stop rules on an SJF-Est base, 600 s cadence, 20% of evaluable jobs declining. Six disjoint three-day windows; stop rules additionally over three seeds. A_{job}, A_{ten}, Q_{10}, A_{gpu} are per-job, per-tenant, 10th-percentile-tenant and GPU-hour-weighted SLO attainment. StopAt-p terminates at a fixed fraction of estimated progress, ignoring the eval signal; EvalStop k is the published k-consecutive-decline rule; LossPlateau stops on a training-loss plateau; Drawdown is Eq. 3.

scheduler	A_{job}	A_{ten}	Q_{10}	A_{gpu}
EDF-SLO	**0.990**	0.984	0.988	0.876
SJF-Est	0.984	0.977	0.956	0.875
DRF-fair	0.980	**0.985**	0.981	**0.901**
SRTF-Est	0.970	0.952	0.852	0.813
LossAware-preempt	0.943	0.912	0.725	0.766
FIFO-backfill	0.931	0.906	0.690	0.859
FIFO	0.563	0.518	0.001	0.819

stop rule	net goodput (GPU-h)	FPR	recall
none	**277,469**	n/a	n/a
StopAt-p	218,165	0.208	0.088
EvalStop k=3	196,305	0.124	0.649
Drawdown c=1.5	180,441	0.185	**0.840**
EvalStop k=2	130,260	0.363	0.789
LossPlateau	119,157	0.388	0.800

no-stop net goodput baseline of 268,444 GPU-hours. Under it, EvalStop $k = 2$ "improves" mean JCT by 16.7% and a naive SLO predicate from 0.9844 to 0.9974, while net goodput falls 44.0% (to 150,435 GPU-hours) and termination-aware per-job attainment falls to 0.872. Freeing capacity helps the survivors, and that single mechanism corrupts several metrics at once. A stop rule *raises* FIFO's per-job attainment (+25.0pp for $k = 2$) while lowering every other scheduler's, so the number is meaningless without naming its base scheduler; above $\rho \approx 1.6$ stop rules begin *helping* per-job attainment; and TTFUC improves too. Net goodput never flips sign at any load, and net goodput reported with the false-positive rate is the only combination that resisted every probe we ran.

Sensitivity. Coarsening the cadence from 300 s to 4,800 s moves every rule's false-positive rate by a factor of 1.2–1.7, shrinks the actionable population from 15,124 to 2,902 jobs, and flips one of our own conclusions, since the drawdown rule's recall advantage over $k = 2$ becomes indistinguishable from zero at the coarsest cadence. Separately, exposing `slo_class` hands a policy a coarsened duration oracle, the classes being derived from duration antimodes, and hiding it degrades EDF-SLO to byte-identical equality with its FIFO-backfill fallback at a cost of 5.56pp of per-job attainment. Any policy consuming that field must be ablated against it (Appendix E).

6.4 A LEARNED POLICY DOES NOT BEAT ONE SCALAR OF ADAPTATION

We trained stopping policies (regularized linear and gradient-boosted) on 19 features available inside the information boundary, splitting by model scale so no trajectory family appears in both fitting and evaluation. The result is negative. At matched recall of 0.90 on held-out scales the best learned variant reaches FPR 0.161 against 0.174 for the tuned one-parameter drawdown baseline, an improvement that is not significant (Wilcoxon $p = 0.068$) and wins in only 4 of 20 cells. Cost-sensitive variants are clearly *worse* on FPR. Learning wins on one axis, detection delay at matched recall (1.54 against 2.50 checkpoints, $p = 1.9 \times 10^{-6}$), bought with false positives.

Underneath that null result sit two findings that carry more weight than a win would have. The first is a **detectability floor**. Median measured SNR rises monotonically with model scale, from -0.168 at 70M and $+0.026$ at 160M, both indistinguishable from zero, to 0.72 at 12B. At the floor the best baseline needs FPR 0.854 to reach even 50% recall, against 0.000 at the three largest scales, so below roughly 0.4 measured SNR no detector we tried separates healthy from declining curves at a

useful operating point. The limit is the observation process, not the policy class. Part of that floor is structural, since measured SNR is $\overline{\Delta}/\mathrm{sd}(\Delta)$ and for a concave curve approaching saturation $\overline{\Delta} \to 0$ by construction, so the two become indistinguishable per interval exactly where the curve flattens, which is also where stopping is worth least. Detection difficulty and the value of detecting fall together, which is why a per-interval decline test is the wrong statistic and cumulative drawdown (Eq. 3) a better one, even though it too falls short. The second finding is that **thresholds do not transfer across noise regimes**. One calibrated for a 10% FPR budget on validation scales buys held-out recall of only 0.40–0.47, with per-fold spread from 0.00 to 0.92, which is the criticism we make of fixed k now applying to our own rule. The leakage ablation is clean: hiding `slo_class` moves the learned policy by 0.02pp of FPR.

7 Limitations and Conclusion

Four limitations most constrain these claims, and Appendix F gives the full treatment. **(i)** The evaluation trajectories are pretraining runs, not RLHF or DPO fine-tuning, the most important limitation of Section 4; what transfers is the dependence on measured SNR, delta structure and checkpoint budget rather than any RLHF-specific SNR value, which is why we report a sweep. **(ii)** All windows come from one trace of a single 2020 cluster that is not an LLM fine-tuning workload; the Acme trace (Hu et al., 2024b) does carry LLM job types, but like every public trace it carries no per-job eval-score trajectories, so no available trace supports this study end to end. **(iii)** Declining trajectories are injected, so recall characterises the shape of the trade-off, not absolute performance. **(iv)** The SLO definition, prevalence, restart overhead and snapshot interval are choices, not measurements.

Within those bounds, scheduler-level early stopping rests on two assumptions we could test against real data, and both fail. A fixed threshold assumes a signal-to-noise regime real evaluation curves do not reach, so it terminates nearly half of known-healthy runs rather than the 1.5% predicted, and standard scheduling metrics are corrupted in turn by the capacity those terminations free. Learning the policy did not fix this, because below roughly 0.4 measured SNR the observation process carries too little signal for any detector. The practical recommendation is a higher evidentiary bar for stopping, and net goodput reported with an FPR quoted at its cadence.

REPRODUCIBILITY STATEMENT

Every quantitative claim in this paper is produced by code run for this submission, and each is traceable to a saved data artifact. The two data sources are public: the Alibaba PAI 2020 GPU cluster trace (Weng et al., 2022) and the published per-checkpoint Pythia evaluation results (Biderman et al., 2023), both downloaded from their official repositories at their documented URLs. Section 5 states the derivation of the workload (filtering, GPU-demand semantics, SLO-class construction) and the simulator's modelling choices and their measured costs. The simulator, all scheduling and stopping policies, the learned policy and its training code, the experiment runner, and the analysis scripts are provided as anonymous supplementary material, together with a test suite of 140 tests that includes a mechanically enforced information-boundary check: any policy reading a field a deployed scheduler could not observe fails the suite. The cell-level result caches for all 819 experiment cells, the 70 offline learning cells, and the 48 in-simulator learning runs are included, so every table and figure can be regenerated without re-running the sweeps. Where a modelling parameter was chosen rather than measured (restart overhead, snapshot interval, declining-job prevalence, checkpoint cadence) we report a sensitivity sweep instead of a single setting.

ETHICS STATEMENT

This work uses two public, non-personal datasets: an anonymized production cluster trace containing job-level resource and timing records with no user content, and published benchmark accuracy numbers for open models. No human subjects, personal data, or new model training on user data are involved. The practical direction of our findings is conservative, in that we argue a published mechanism for automatically terminating other tenants' training jobs fires far more often on healthy runs than reported, and that several standard metrics conceal the resulting damage. Acting on the metrics we criticise could cause a platform to destroy tenants' work while its dashboards improved, which is the risk this paper is intended to reduce. Our own learned policy is reported as a negative result precisely so that it is not adopted on the strength of an apparent but non-significant improvement.

AI USE STATEMENT

In this work, we used generative AI tools for the following tasks with **required disclosure**: proposing and refining hypotheses; designing and providing feedback on research methodology and experiments; implementing methods; generating synthetic data sets; and interpreting results. We have not used generative AI tools for developing theoretical models, formulating mathematical claims, providing ingredients for or assisting in the writing of proofs, translation, or supporting qualitative and thematic data analysis; the remaining required-disclosure tasks are not applicable to this work. Additionally, we used generative AI tools for these tasks with **recommended disclosure**: sourcing and searching for information; identifying relevant literature; formatting references; creating and modifying scientific figures; creating and editing software code; suggesting experimental parameters; drafting parts of the paper; and proposing its structure and title. We have reviewed all AI-assisted work and take responsibility for the final content of this work, including text, claims and artifacts produced with the aid of generative AI.

We elaborate, because the extent of AI involvement here is substantial and the disclosure would be misleading if compressed. The research programme was directed by the authors, who set the topic, the venue, the constraints and the standards of evidence, and who adjudicated the major framing decisions. Within that direction, an AI system carried out the literature and data reconnaissance, wrote essentially all of the analysis and simulation code, executed the experiments, produced the figures, and drafted the text.

Three consequences deserve specific mention. First, *hypothesis revision*: the paper's original framing was materially changed after the cited prior work was checked against its actual text and found to have been misdescribed, and after a numerical claim carried in the project's working notes could not be verified against any source; the claim was dropped and the framing replaced, with the authors selecting among the alternatives presented. Second, *synthetic data*: synthetic evaluation curves were generated to re-implement the predecessor's stated healthy-curve model for comparison against real curves, and synthetic declines were grafted onto real trajectories to measure detector recall, since no

ground-truth corpus of reward-hacking runs exists. Both are identified as synthetic wherever they are used. Third, *self-correction*: several of this paper's reported findings refute hypotheses stated at the outset of the work, including the attribution of the fixed-threshold failure, the expected ranking inversion between latency and SLO objectives, and the expectation that abrupt declines would favour fixed thresholds. Several accounting errors in intermediate analyses (crediting re-run work as goodput, an unbalanced sweep comparison, a degenerate optimum in a training objective) were also found and fixed during the work, and each is documented in the supplementary reports instead of being silently corrected.

Verification was performed as follows. Claims attributed to prior work were read directly from the source document rather than recalled. Every bibliography entry was resolved against a bibliographic API, a publisher record, or the data provider's own requested citation. Dataset field semantics were confirmed against the provider's official documentation rather than assumed. Numeric claims in the text were checked programmatically against the generating data artifacts, and an independent automated review process ran throughout, flagging errors that were then corrected and re-verified. We set this out because it is an accurate account of how the paper was produced, and because the authors remain responsible for anything the checks missed.

REFERENCES

Saurabh Agarwal, Amar Phanishayee, and Shivaram Venkataraman. Blox: A modular toolkit for deep learning schedulers. In *Proceedings of the Nineteenth European Conference on Computer Systems (EuroSys)*, 2024. doi: 10.1145/3627703.3629583.

Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, et al. Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning*, pp. 2397–2430. PMLR, 2023.

Georgia Christofidi, Konstantinos Papaioannou, and Thaleia Dimitra Doudali. Is machine learning necessary for cloud resource usage forecasting? In *Proceedings of the 2023 ACM Symposium on Cloud Computing (SoCC)*, 2023. doi: 10.1145/3620678.3624790.

Mostafa Dehghani, Yi Tay, Alexey A. Gritsenko, Zhe Zhao, Neil Houlsby, Fernando Diaz, Donald Metzler, and Oriol Vinyals. The benchmark lottery. *arXiv preprint arXiv:2107.07002*, 2021.

Tobias Domhan, Jost Tobias Springenberg, and Frank Hutter. Speeding up automatic hyperparameter optimization of deep neural networks by extrapolation of learning curves. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 3460–3468, 2015.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. A framework for few-shot language model evaluation, September 2021.

Leo Gao, John Schulman, and Jacob Hilton. Scaling laws for reward model overoptimization. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202, pp. 10835–10866, 2023.

Daniel Golovin, Benjamin Solnik, Subhodeep Moitra, Greg Kochanski, John Karro, and D. Sculley. Google vizier: A service for black-box optimization. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pp. 1487–1495, 2017. doi: 10.1145/3097983.3098043.

Juncheng Gu, Mosharaf Chowdhury, Kang G. Shin, Yibo Zhu, Myeongjae Jeon, Junjie Qian, Hongqiang Liu, and Chuanxiong Guo. Tiresias: A GPU cluster manager for distributed deep learning. In *16th USENIX Symposium on Networked Systems Design and Implementation (NSDI 19)*, pp. 485–500, 2019.

Torsten Hoefler and Roberto Belli. Scientific benchmarking of parallel computing systems: Twelve ways to tell the masses when reporting performance results. In *Proceedings of the International*

Conference for High Performance Computing, Networking, Storage and Analysis (SC), 2015. doi: 10.1145/2807591.2807644.

Steven R. Howard, Aaditya Ramdas, Jon McAuliffe, and Jasjeet Sekhon. Time-uniform, nonparametric, nonasymptotic confidence sequences. *The Annals of Statistics*, 49(2):1055–1080, 2021. doi: 10.1214/20-AOS1991.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.

Jian Hu, Xibin Wu, Zilin Zhu, Xianyu, Weixun Wang, Dehao Zhang, and Yu Cao. OpenRLHF: An easy-to-use, scalable and high-performance RLHF framework. *arXiv preprint arXiv:2405.11143*, 2024a.

Qinghao Hu, Zhisheng Ye, Meng Zhang, Qiaoling Chen, Peng Sun, Yonggang Wen, and Tianwei Zhang. Hydro: Surrogate-based hyperparameter tuning service in datacenters. In *17th USENIX Symposium on Operating Systems Design and Implementation (OSDI 23)*, pp. 757–777, 2023.

Qinghao Hu, Zhisheng Ye, Zerui Wang, Guoteng Wang, Meng Zhang, Qiaoling Chen, Peng Sun, Dahua Lin, Xiaolin Wang, Yingwei Luo, Yonggang Wen, and Tianwei Zhang. Characterization of large language model development in the datacenter. In *21st USENIX Symposium on Networked Systems Design and Implementation (NSDI 24)*, pp. 709–729, 2024b.

Sungjin Im, Ravi Kumar, Mahshid Montazer Qaem, and Manish Purohit. Non-clairvoyant scheduling with predictions. In *Proceedings of the 33rd ACM Symposium on Parallelism in Algorithms and Architectures (SPAA)*, pp. 285–294, 2021. doi: 10.1145/3409964.3461790.

Kevin Jamieson and Ameet Talwalkar. Non-stochastic best arm identification and hyperparameter optimization. In *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 51 of *Proceedings of Machine Learning Research*, pp. 240–248, 2016.

Suhas Jayaram Subramanya, Daiyaan Arfeen, Shouxu Lin, Aurick Qiao, Zhihao Jia, and Gregory R. Ganger. Sia: Heterogeneity-aware, goodput-optimized ML-cluster scheduling. In *Proceedings of the 29th Symposium on Operating Systems Principles (SOSP)*, pp. 642–657, 2023. doi: 10.1145/3600006.3613175.

Myeongjae Jeon, Shivaram Venkataraman, Amar Phanishayee, Junjie Qian, Wencong Xiao, and Fan Yang. Analysis of large-scale multi-tenant GPU clusters for DNN training workloads. In *2019 USENIX Annual Technical Conference (USENIX ATC 19)*, pp. 947–960, 2019.

Aaron Klein, Stefan Falkner, Jost Tobias Springenberg, and Frank Hutter. Learning curve prediction with bayesian neural networks. In *International Conference on Learning Representations*, 2017.

Liam Li, Kevin Jamieson, Afshin Rostamizadeh, Ekaterina Gonina, Jonathan Ben-tzur, Moritz Hardt, Benjamin Recht, and Ameet Talwalkar. A system for massively parallel hyperparameter tuning. In *Proceedings of Machine Learning and Systems*, volume 2, 2020.

Lisha Li, Kevin Jamieson, Giulia DeSalvo, Afshin Rostamizadeh, and Ameet Talwalkar. Hyperband: A novel bandit-based approach to hyperparameter optimization. *Journal of Machine Learning Research*, 18(185):1–52, 2018.

G. Lorden. Procedures for reacting to a change in distribution. *The Annals of Mathematical Statistics*, 42(6):1897–1908, 1971. doi: 10.1214/aoms/1177693055.

Hongzi Mao, Malte Schwarzkopf, Shaileshh Bojja Venkatakrishnan, Zili Meng, and Mohammad Alizadeh. Learning scheduling algorithms for data processing clusters. In *Proceedings of the ACM Special Interest Group on Data Communication (SIGCOMM)*, pp. 270–288, 2019. doi: 10.1145/3341302.3342080.

Jeffrey C. Mogul and John Wilkes. Nines are not enough: Meaningful metrics for clouds. In *Proceedings of the Workshop on Hot Topics in Operating Systems (HotOS)*, pp. 136–141, 2019. doi: 10.1145/3317550.3321432.

Ted Moskovitz, Aaditya K. Singh, DJ Strouse, Tuomas Sandholm, Ruslan Salakhutdinov, Anca Dragan, and Stephen McAleer. Confronting reward model overoptimization with constrained RLHF. In *The Twelfth International Conference on Learning Representations*, 2024.

Rajeev Motwani, Steven Phillips, and Eric Torng. Nonclairvoyant scheduling. *Theoretical Computer Science*, 130(1):17–47, 1994. doi: 10.1016/0304-3975(94)90151-1.

Deepak Narayanan, Keshav Santhanam, Fiodar Kazhamiaka, Amar Phanishayee, and Matei Zaharia. Heterogeneity-aware cluster scheduling policies for deep learning workloads. In *14th USENIX Symposium on Operating Systems Design and Implementation (OSDI 20)*, pp. 481–498, 2020.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

E. S. Page. Continuous inspection schemes. *Biometrika*, 41(1–2):100–115, 1954. doi: 10.1093/biomet/41.1-2.100.

Lutz Prechelt. Early stopping — but when? In Genevieve B. Orr and Klaus-Robert Müller (eds.), *Neural Networks: Tricks of the Trade*, volume 1524 of *Lecture Notes in Computer Science*, pp. 55–69. Springer, Berlin, Heidelberg, 1998. doi: 10.1007/3-540-49430-8_3.

Manish Purohit, Zoya Svitkina, and Ravi Kumar. Improving online algorithms via ML predictions. In *Advances in Neural Information Processing Systems*, volume 31, pp. 9661–9670, 2018.

Aurick Qiao, Sang Keun Choe, Suhas J. Subramanya, Willie Neiswanger, Qirong Ho, Hao Zhang, Gregory R. Ganger, and Eric P. Xing. Pollux: Co-adaptive cluster scheduling for goodput-optimized deep learning. In *15th USENIX Symposium on Operating Systems Design and Implementation (OSDI 21)*, pp. 1–18, 2021.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

Rafael Rafailov, Yaswanth Chittepu, Ryan Park, Harshit Sikchi, Joey Hejna, W. Bradley Knox, Chelsea Finn, and Scott Niekum. Scaling laws for reward model overoptimization in direct alignment algorithms. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

Ying Sheng, Shiyi Cao, Dacheng Li, Coleman Hooper, Nicholas Lee, Shuo Yang, Christopher Chou, Banghua Zhu, Lianmin Zheng, Kurt Keutzer, Joseph E. Gonzalez, and Ion Stoica. S-LoRA: Serving thousands of concurrent LoRA adapters. In *Proceedings of Machine Learning and Systems*, volume 6, 2024.

Joar Skalse, Nikolaus H. R. Howe, Dmitrii Krasheninnikov, and David Krueger. Defining and characterizing reward hacking. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

Alexander Tartakovsky, Igor Nikiforov, and Michèle Basseville. *Sequential Analysis: Hypothesis Testing and Changepoint Detection*. Chapman & Hall/CRC, 2014.

Qizhen Weng, Wencong Xiao, Yinghao Yu, Wei Wang, Cheng Wang, Jian He, Yong Li, Liping Zhang, Wei Lin, and Yu Ding. MLaaS in the wild: Workload analysis and scheduling in large-scale heterogeneous GPU clusters. In *19th USENIX Symposium on Networked Systems Design and Implementation (NSDI 22)*, pp. 945–960, 2022.

Guilin Zhang, Chuanyi Sun, Kai Zhao, Xu Chu, Shahryar Sarkani, and John M. Fossaceca. Evalstop: Using world feedback to detect and correct reward overoptimization in multi-tenant RLHF platforms. *arXiv preprint arXiv:2606.04145*, 2026.

Haoyu Zhang, Logan Stafman, Andrew Or, and Michael J. Freedman. SLAQ: Quality-driven scheduling for distributed machine learning. In *Proceedings of the 2017 Symposium on Cloud Computing (SoCC)*, pp. 390–404, 2017. doi: 10.1145/3127479.3127490.

Pengfei Zheng, Rui Pan, Tarannum Khan, Shivaram Venkataraman, and Aditya Akella. Shockwave: Fair and efficient cluster scheduling for dynamic adaptation in machine learning. In *20th USENIX Symposium on Networked Systems Design and Implementation (NSDI 23)*, pp. 703–723, 2023.

A Evaluation trajectories: task admission and diagnostics

Which tasks we admit. Of the 65 tasks with published per-checkpoint results for the Pythia suite (Biderman et al., 2023; Gao et al., 2021) we admit 6. We exclude the 57 MMLU subjects, which never leave a band around the four-way chance level, and `logiqa` and `wsc`, which show no learning signal at any model scale. Trajectories that are pure noise around chance would inflate our false-positive rate for the wrong reason. This leaves 78 model $\times$ task trajectories over 13 model scales.

Why a 14-checkpoint tail. Pythia's 27-step checkpoint grid is logarithmically spaced early in training, whereas a scheduler observes evenly spaced checkpoints. We therefore evaluate a uniform 14-checkpoint tail. The choice is conservative: the full grid gives a substantially higher false-positive rate than the 44.9% we report.

Per-decision rather than per-job rates. Section 4 reports per-job (per-trajectory) false-positive rates throughout, matching the published figure's denominator. Measured per decision instead, the $k = 2$ rule's rate is 5.52% over $1{,}014$ decisions.

Nominal versus measured SNR. The two diverge sharply for a saturating mean, where a nominal SNR of 4.0 realises as a measured 1.14, so quoting one for the other overstates the attainable regime by 3–4$\times$. This is why we report measured SNR throughout.

Dependence structure of the deltas. Beyond the run-length excesses in Section 4, the successive deltas carry lag-1 autocorrelation -0.115 ($p = 4.6 \times 10^{-3}$), excess kurtosis $+0.70$, and Jarque–Bera $p = 6.2 \times 10^{-14}$. The departure from the i.i.d. Gaussian calibration model is therefore in the dependence and the tails, not only in the scale.

B The quickest-detection bound, and what it does not say

Section 4 uses Lorden's minimax bound (Lorden, 1971) to argue that the published operating point is unreachable at the measured signal-to-noise regime. The arithmetic is as follows. Lorden gives the optimal detector's worst-case expected delay as asymptotically $\log \gamma / D$, where γ is the average run length to false alarm and D the Kullback–Leibler divergence per observation between the post- and pre-change laws. For a Gaussian mean shift of size δ against noise σ, $D = \delta^2/2\sigma^2 = \mathrm{SNR}^2/2$, so the delay scales as $2 \log \gamma / \mathrm{SNR}^2$. Treating false alarms over a horizon of $n = 14$ checkpoints as approximately geometric with mean γ, a per-trajectory false-alarm probability of 1.5% requires $\gamma \approx n/0.015 \approx 930$; at the median measured $\mathrm{SNR} = 0.55$ this gives $2 \log(930)/0.55^2 \approx 45$ checkpoints of delay, against the 14 available.

Two assumptions keep this an order-of-magnitude statement and not a theorem about our setting. We flag both, since the second is one our own data violates. First, identifying the healthy-curve drift SNR with a post-change mean shift of equal magnitude is an idealisation, and the true declining-curve dynamics are not a step change in the mean. Second, Lorden's bound in its classical form assumes independent observations, and Section 4 reports lag-1 autocorrelation of -0.115 and excess kurtosis of $+0.70$ in the successive deltas. Non-i.i.d. forms of the delay/false-alarm relation exist (Tartakovsky et al., 2014), and a threshold set from an anytime-valid boundary (Howard et al., 2021) would make the error budget a stated parameter instead of an observed outcome. We did not pursue either, and a tighter treatment of the bound under the measured dependence structure is the most obvious extension of this diagnosis. Both effects reduce the effective information per observation, which lengthens the required delay, so the direction of the argument holds even where the constant does not.

C TRACE CHARACTERISATION

Section 5 states three properties of the Alibaba PAI 2020 workload that contradict the synthetic setup the predecessor calibrated against. The supporting tests are these.

Arrivals are not Poisson. A Kolmogorov–Smirnov test against an exponential inter-arrival law gives $D = 0.156$ ($p < 10^{-300}$). The index of dispersion rises from 15.2 at a $60\,\mathrm{s}$ bin to 1401 at a one-day bin, and the Hurst exponent is 0.75. The cause is batch submission rather than diurnal structure: 36.4% of jobs share their arrival second, and conditioning on hour-of-day still leaves a median within-hour $D = 3.8$.

Durations have decreasing hazard. A lognormal fit is best by AIC ($\sigma = 2.11$) and the Weibull shape parameter is $c = 0.48 < 1$, so elapsed runtime positively predicts remaining time. This is adverse for any remaining-time heuristic, including the base schedulers these stop rules compose with.

Demand is concentrated and fractional. The Gini coefficient on job counts is 0.930 and 37.4% of jobs request less than one GPU. The tail is load-bearing: the 0.18% of jobs requesting more than 64 GPUs hold 9.7% of all GPU-hours, and the longest 1% hold 35.7% of GPU-seconds, which is why we do not trim it.

Class assignment and capacity. Class is assigned per job rather than per tenant because the median tenant has only 70% of its jobs in its own dominant class, falling to 59% for the 50 largest tenants. Capacity is calibrated from the trace's own occupancy (mean $2{,}881.4$, peak $5{,}218.2$ concurrent GPUs). With capacity removed, all $714{,}903$ jobs complete at exactly $a_j + S_j$ with a maximum error of $0\,\mathrm{s}$.

D OBJECTIVE DIAGNOSTICS

Section 3 reports that we expected mean JCT to misrank policies against SLO attainment and that it does not. Across the policy set, Kendall's τ between the two rankings is 0.82, the material inversion rate is 5.3%, and the JCT-optimal policy forfeits a mean of only 0.19pp of attainment. Mean JCT is insensitive rather than wrong. Since $\mathrm{JCT}_j = w_j + S_j$ with S_j schedule-invariant, only $\overline{w}^{\pi}$ is controllable, and the policy-invariant service term is 96.6% of mean JCT at $\rho = 0.9$ and still 44.7% at $\rho = 2.0$. TTFUC behaves the same way in absolute terms, where every policy reads 134–140ks because the quantity is $\approx 95\%$ arrival time, which is why Section 3 measures it relative to submission.

E ADDITIONAL SENSITIVITY RESULTS

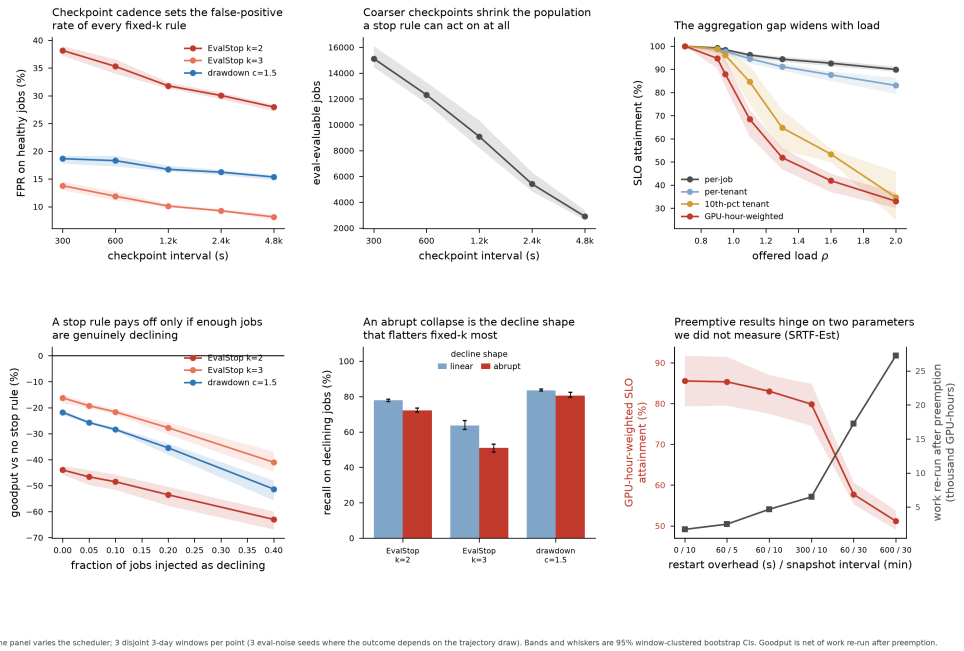
THE REQUIRED SENSITIVITY: CHECKPOINT CADENCE SETS EVERY FPR

Coarsening the evaluation cadence from $300\,\mathrm{s}$ to $4{,}800\,\mathrm{s}$, $k = 2$'s false-positive rate moves $0.382 \to 0.280$ (a factor of 1.36), $k = 3$'s $0.138 \to 0.082$, and the drawdown rule's $0.187 \to 0.154$. A reported detector operating point without its cadence is not a well-defined number. Coarsening the cadence also shrinks the population a stop rule can act on at all, from $15{,}124$ to $2{,}902$ evaluable jobs.

This sweep flips one of our own conclusions: the drawdown rule's recall advantage over $k = 2$ shrinks roughly fourfold across the range ($+0.068$ at $300\,\mathrm{s}$ to -0.024 at $4{,}800\,\mathrm{s}$) and at the coarsest cadence the two are no longer distinguishable (CI $[-0.053, +0.002]$). Its false-positive advantage survives everywhere (19.51pp at $300\,\mathrm{s}$ to 12.62pp at $4{,}800\,\mathrm{s}$, all CIs excluding zero).

THE BEST PER-JOB SCHEDULER OWES ITS POSITION TO AN INFORMATION LEAK

Because SLO classes are derived from duration antimodes, exposing `slo_class` to a policy hands it a coarsened duration oracle. Hiding it degrades EDF-SLO to its FIFO-backfill fallback, and its metrics become byte-identical to FIFO-backfill's ($A_{\text{job}} = 0.9345$, $Q_{10} = 0.7029$, p_{95} wait $1{,}309.5\,\mathrm{s}$



SJF-Est base unless the panel varies the scheduler; 3 disjoint 3-day windows per point (3 eval-noise seeds where the outcome depends on the trajectory draw). Bands and whiskers are 95% window-clustered bootstrap CIs. Goodput is net of work re-run after preemption.

Figure 3: Sensitivity sweeps. SJF-Est base unless the panel varies the scheduler; three disjoint three-day windows per point, bands are 95% window-clustered bootstrap CIs, goodput net of work re-run after preemption.

in both). The cost of hiding the field is -5.56pp $[-7.78, -3.49]$ per-job attainment and -28.69pp $[-43.4, -14.8]$ for the 10th-percentile tenant. The leak is not uniformly beneficial, since SRTF-Est is *hurt* on the GPU-hour-weighted metric (-1.55pp), but any policy that consumes this field, learned policies especially, must be ablated against it.

Simulator validation

Learned policy operating curves

F Extended limitations

Evaluation trajectories are from pretraining, not fine-tuning. This is the most important limitation of Section 4. Pythia runs are ordinary pretraining, which is what makes them clean negatives, but they are not RLHF or DPO fine-tuning runs. What transfers is the statistical claim: a fixed-k false-positive rate is a function of the measured SNR regime, the dependence structure of successive deltas, and the checkpoint budget. What does not transfer is any RLHF-specific SNR value, which is why we report a sweep instead of a single headline number.

One trace, six windows; injected declines. All windows come from a single 68.4-day trace of one 2020 production cluster that is not an LLM fine-tuning workload, so six windows are six bootstrap units (three for the sweeps), confidence intervals are wide, and we do not correct for multiple comparisons across the roughly 120-row significance table. Because no public corpus of ground-truth reward-hacking runs exists, recall is measured against synthetic declines grafted onto real curves; recall numbers characterise the shape of the trade-off, not absolute detector performance. Read every generalisation as conditional on this trace's structure.

Modelling choices without empirical grounding. The SLO definition, the 20% default declining fraction, the restart overhead and the snapshot interval are choices, not measurements. The last two matter. Across plausible settings net goodput is invariant but GPU-hour-weighted attainment swings

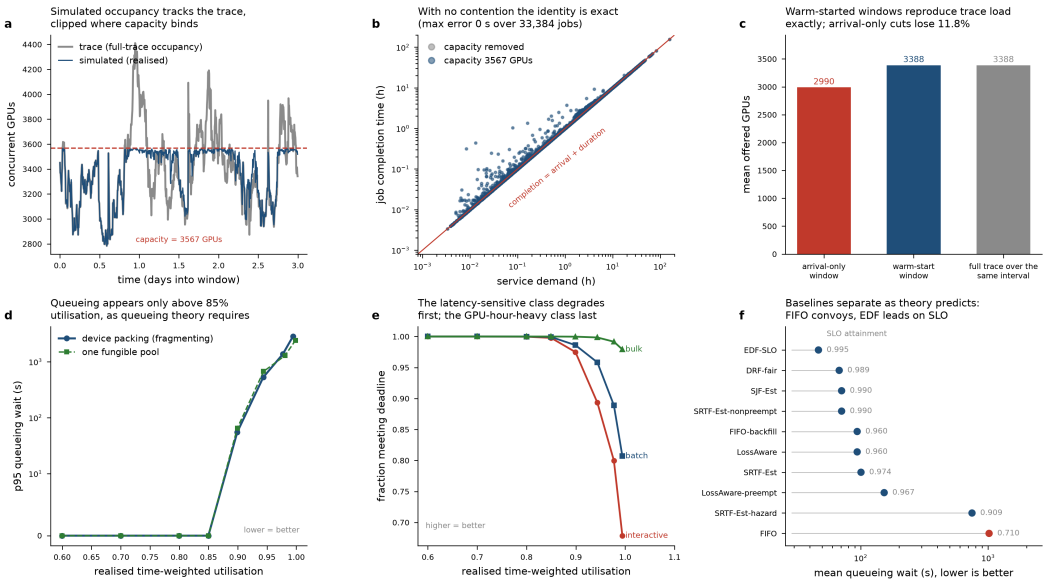


Figure 4: Simulator validation. (a) Simulated occupancy tracks the trace, clipped where capacity binds. (b) With capacity removed the completion identity is exact over all jobs. (c) Warm-started windows reproduce full-trace offered load exactly; an arrival-only cut loses 11.8%. (d) Queueing appears only above 85% realised utilisation. (e) The latency-sensitive class degrades first. (f) Baselines separate as scheduling theory predicts.

34pp and re-run work varies 16×, so our preemptive results are conditional on them and we report the dependence instead of a single setting.

Scope, and what we did not do. At a 600 s cadence only about a third of jobs are eval-evaluable and that population is essentially entirely bulk-class, so nothing here supports a claim about interactive or batch jobs (67.7% of jobs, 3.95% of GPU-hours). The bulk class, conversely, holds 96.05% of GPU-hours. We quote the predecessor's figures as claims that paper makes, since we did not re-run its experiment and make no claim its reported JCT gain is an artifact, though our probe shows mean JCT is capable of producing one. The learning-side contribution is a stopping rule trained with a cost-sensitive contextual objective, not a reinforcement-learning agent. Since a linear rule matches a gradient-boosted one and neither beats a one-parameter baseline, we judged further machinery unlikely to change the verdict. That is a judgement, not an experiment.

No real-hardware validation

Every result in this paper is produced by trace-driven simulation. We did not run a scaled-down fine-tuning testbed on physical GPUs, so we have no measurement of how far the simulator's predictions track a real multi-tenant deployment. Three specific gaps follow. First, the execution model is non-preemptive with no slowdown, so service demand is schedule-invariant by construction. On real hardware, co-located jobs interfere and that identity would not hold exactly. Second, restart overhead and snapshot interval are parameters we sweep rather than quantities we measured, and the sweep shows they move GPU-hour-weighted attainment by 34pp, so a measured value would meaningfully narrow the preemptive results. Third, the evaluation trajectories are replayed from published checkpoints rather than generated by the simulated jobs themselves, which is what makes them real but also means eval cost and eval-induced contention are absent from the model. The simulator is validated against the trace's own ground truth (Appendix E) and its in-simulation false-positive rates reconcile with the offline diagnosis to within 0.0005, but neither check substitutes for hardware measurement.

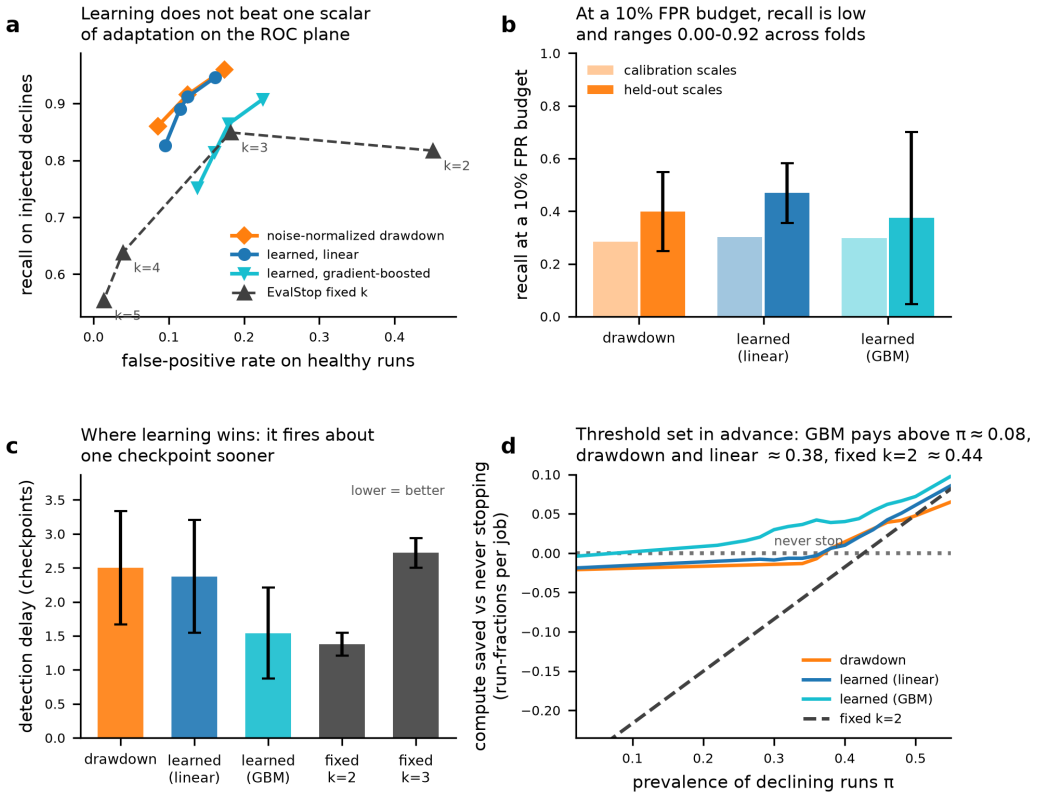


Figure 5: Learned stopping policy against the fixed-k family and the noise-normalized drawdown baseline. (a) On the ROC plane, learning does not beat one scalar of adaptation. (b) At a 10% FPR budget, held-out recall is low and varies widely across folds, so thresholds do not transfer across noise regimes. (c) Learning's one advantage is detection delay at matched recall. (d) Compute saved against the prevalence of declining runs; each rule's break-even crossing is named. At this trace's prevalence ($\pi = 0.198$) every rule is below its own break-even.