To Forge or Not to Forge: Predicting Task-Level Fine-Tuning Gains from Ten-Example Pilots

Tangyi Qian
Independent Researcher

Abstract

Fine-tuning a small specialist is the default path to cheap task-specific capability, yet the upstream decision — is this task worth fine-tuning at all? — is still made by intuition or by paying for the full run. Under one frozen protocol (Qwen2.5-1.5B-Instruct, fixed LoRA recipe and budgets, greedy decoding, execution-based verifiers) we pair a ten-example pilot with a full fine-tune on 61 tasks — 11 anchored in the small-model post-training literature, 50 drawn by a pre-registered filter from Super-NaturalInstructions — and take the measured gain as ground truth. Fine-tuning helps on 53 of 61 tasks (87%), but 23 tasks start at pass@1 $= 0$, where the gain is format unlock rather than new capability. The pilot *ranks* gain magnitude (Spearman 0.755, CI $[0.60, 0.86]$), and the ranking has budget value: spending K=20 full fine-tunes in pilot order captures 0.72 of all positive-gain mass, against 0.33 random and 0.74 oracle. The pre-registered binary go/no-go endpoint is published as a negative (leave-one-task-out AUC 0.755, CI $[0.50, 0.93]$). A declared revision locates the mechanism: removing the one post-decision feature does *not* rescue it (0.663, CI $[0.43, 0.84]$), while a three-feature pre-decision model does (0.767, CI $[0.65, 0.88]$, permutation p=0.004) — the registered negative was an over-parameterised estimator, not an absence of signal. Binary framing was in any case the wrong target: a *perfect* gate is worth 2.4% over always forging, and the probe is not free (median 0.17 of full training-plus-eval cost, 0.53 including the base evaluation it needs). Instruction embeddings predict at chance (0.512): you have to run the probe. We characterise the no-go tail, including negative transfer on heavily post-trained bases (GSM8K -0.23 despite in-domain training), and archive the benchmark, protocol, analysis code, and all sealed artifacts. Every measurement here was produced by an autonomous discover–pilot–decide–train–validate system; without it the 61-task ground-truth table would not exist.

Keywords: fine-tuning, evaluation, small language models

1 Autonomous research result

1.1 Hypothesis

Under a single frozen LoRA recipe on a 1.5B instruction-tuned base, can a ten-example probe decide *which* tasks to fully fine-tune, and is that decision a binary gate or a ranking for a budget of K slots? We pre-register a binary label $\mathrm{go} \Leftrightarrow \Delta_{\mathrm{full}} > 0$ and a secondary ranking endpoint $\rho(\Delta_{\mathrm{pilot}}, \Delta_{\mathrm{full}})$, with the honesty clause that a non-significant AUC is published as a negative. A later, declared revision (v2) asks the same questions after removing post-decision features from the classifier, and quantifies the *utility* of a perfect gate versus always forging. A correction to that revision (the addendum) separates the effect of removing leakage from the effect of one declared bug fix; we report both.

1.2 Method

Protocol and sample. Base model Qwen2.5-1.5B-Instruct [Qwen Team, 2024], weights pinned at birth; LoRA rank 16, $\alpha = 32$; sequence 4096; effective batch 16; pilot 10×100; full cap 8000 / 3 epochs; eval 200, greedy. pass@8 (k=8) is a signal (1600 generations per task), never the label. The sealed sample is n=61 (11 literature-layer + 50 SuperNI, seed 20260822): 53 registered go / 8 no-go (87%). BIRD and APPS are out, `excluded for logistics, before any label existed`. All numbers below that are not marked "registered" come from `ANALYSIS_RESULTS_v2` and its correction file `ANALYSIS_RESULTS_v2_ADDENDUM` (CPU revision; the addendum supersedes the v2 attribution wherever they differ). Registered AUC/CI remain the v1 sealed row.

Predictors. The *new main* classifier is leave-one-task-out logistic regression on the pre-decision vector only: Δ_{pilot}, base pass@1, base pass@8, headroom, pilot-loss descriptors, format-compliance of the *base* split, and train_n. $\text{gen\_len.full.median}$ is reported as a post-hoc oracle upper bound and is *not* used for decisions. The registered 12-feature vector (which included that oracle and full_n) is kept as the registered row. Intervals are task-level bootstrap 95% CIs (seed 20260820, B=1000). A label-permutation test re-runs full LOTO N=1000 times (seed 20260826) and is reported as $(1 + \#\{\rho_{\text{perm}} \geq \rho_{\text{obs}}\})/(1 + N)$.

One declared bug fix, found after the revision contract. The revision code also changed one feature *definition*: the v1 format-compliance scalar $1 - \text{unparseable}/n$ silently returns 1.0 for any task whose compliance block carries no `unparseable` key. Exactly two of the 61 tasks are like that — GSM8K, which v1 special-cased, and MATH, which it missed. The revision adds the matching MATH branch ($\text{boxed}/n = 0.55$). This is a bug fix, not a task-specific choice, but it was found *after* `ANALYSIS_PREREG_V2` was written, it is not one of that page's declared items, and it lands on one of only two substantive negatives. We therefore report every affected number under both definitions and never fold its effect into "removing leakage" (Table 1). We also report L2-CV LOTO, a 3-feature model ($\Delta_{\text{pilot}}, \text{base\_pass1}, \text{headroom}$), and a no-pass@8 feature set. Instruction embeddings (MiniLM, CPU) plus SuperNI Categories/Source are the semantic control; the original four scalars are the metadata control. McNemar exact tests on paired eval jsonl induce a three-way label (go / no-go / undetermined) as a revision-period analysis; the registered label stays $\Delta_{\text{full}} > 0$. Sensitivity (b) (drop EM-floor rows) is computed but *not* used as supporting evidence (wide CI).

Related work (lite). Training-free transferability estimators (LogME, LEEP, TransRate) score a frozen representation against target labels without fine-tuning [You et al., 2021, Nguyen et al., 2020, Huang et al., 2021]. We did *not* run them: they select among source models, whereas we select among *tasks* given one frozen recipe. That is a limitation and a listed future comparison. Multi-fidelity methods (successive halving, Hyperband, learning-curve extrapolation) [Jamieson and Talwalkar, 2016, Li et al., 2018] treat cheap runs as a fidelity axis. Our probe is a *single* fidelity point; a fidelity scan is future work.

1.3 Evidence

Finding 1: forging is the default *in this regime*. 53 of 61 tasks (87%) have $\Delta_{\text{full}} > 0$. This is Qwen2.5-1.5B-Instruct, one LoRA recipe, greedy pass@1, gate $\Delta_{\text{full}} > 0$ — not a claim about 7B models or full FT. 23 of 61 tasks have $\text{base\_pass1} = 0$; most of those gains are format unlock (the probe teaches the extractor), not new reasoning.

Finding 2: the probe ranks gain, and the ranking has budget value. Spearman $\rho(\Delta_{\text{pilot}}, \Delta_{\text{full}}) = 0.755$, CI $[0.595, 0.864]$ (registered secondary; excludes 0). Partial ρ controlling for base_pass1 is 0.832 (residualise the raw deltas on base, then rank; the fully rank-based partial formula gives 0.777 — either way the ranking is not a restatement of base accuracy). Figure 1 is the operational plot. Ranking by Δ_{pilot} and spending a budget of K full fine-tunes captures 0.42 / 0.72 / 0.83 of all positive-gain mass at K=10, 20, 30, against random 0.16 / 0.33 / 0.49 and oracle 0.45 / 0.74 / 0.90. At K=20 the probe is within three points of oracle. 38% of tasks have $|\Delta_{\text{pilot}}| \leq 0.02$ (a dumb probe); the ranking still works because the mass sits in the tails. Figure 2 shows the same pairing; GSM8K/MATH sit in Q3 and the EM-floor quartet sits at the origin.

Finding 3: the binary gate, three ways. *Registered row.* LOTO on the original 12-feature vector (including post-decision gen_len.full.median) has AUC 0.755, CI $[0.500, 0.929]$, covering 0.5 — a negative under the honesty clause. (The point 0.7547 and Spearman 0.7553 both print as 0.755; that is a rounding coincidence, not one statistic.) *Revision rows, decomposed.* Dropping the post-decision features *lowers* the AUC to 0.663, CI $[0.433, 0.839]$, permutation p=0.057 — still a negative. The jump to 0.844 $[0.733, 0.943]$, p=0.002, comes from the declared MATH format-compliance fix above: one cell of the 61×10 design matrix, worth +0.182 AUC on its own. The leaked feature was *helping* the registered row, so the v1 negative cannot be read as "leakage." It is read correctly as *over-parameterisation*: 12 features against 8 negatives. The parsimonious pre-decision model — three features, no format-compliance term, and therefore invariant to both the leak and the fix — is significant at 0.767 $[0.646, 0.879]$, p=0.004. That row, not 0.844, is the defensible pre-decision estimate. L2-CV (0.875) and the no-pass@8 set (0.837) inherit the same +0.15 dependence on the fix and are reported with their unfixed values in Table 1. None of these rows is stable at the second decimal: dropping a single substantive negative moves the 0.844 model to 0.685 (MATH) or 0.714 (GSM8K), across a jackknife range $[0.685, 0.871]$. *Utility.* A perfect gate's upside versus always-forging is $|\sum \text{negative } \Delta| / \sum \text{positive } \Delta = 0.024$. Net always-forge is +16.6 pass points; oracle-forge is +17.0. McNemar three-way labels are 45 go / 2 no-go / 14 undetermined (α=0.05, uncorrected). With two significant negatives the determined-only AUC is not a serious gate estimate. Figure 3 is the pre-decision ROC.

Finding 4: you still have to run the probe. Instruction-sentence embeddings plus SuperNI category/source have AUC 0.512 $[0.255, 0.746]$ — chance. The old four-scalar "text-only" AUC 0.309 is a *metadata* control whose eval_n is constantly 200; it is not evidence that description features run below chance in any semantic sense. Single-feature steps_to_0.01 is 0.748 $[0.585, 0.881]$ (does not cover 0.5; five tests, *not* multiplicity-corrected). gen_len.full.median is 0.818 $[0.528, 0.978]$ and is post-hoc.

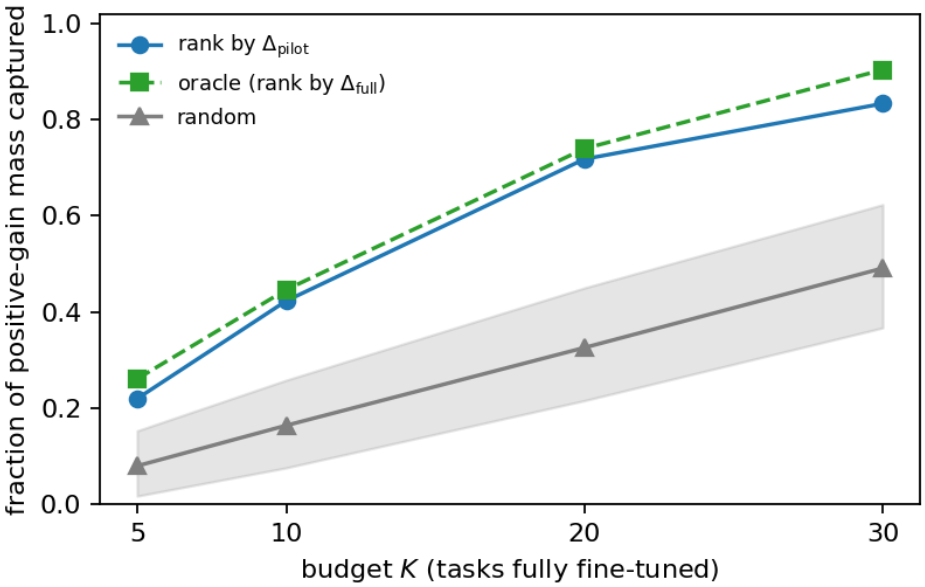

Figure 1: Budget allocation. K full fine-tunes, fraction of positive Δ_{full} mass captured.

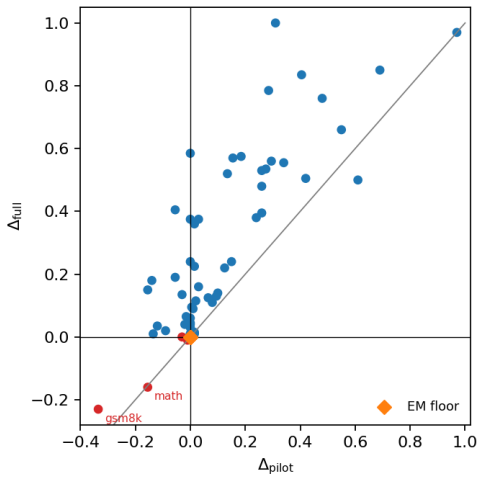

Figure 2: Pilot vs. full, cropped. Diamonds: EM-floor rows.

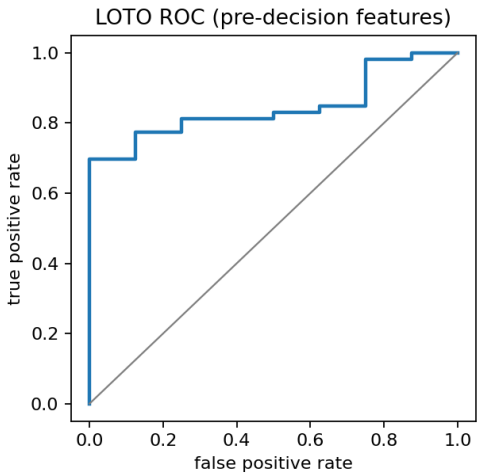

Figure 3: LOTO ROC, pre-decision features.

Table 1: Classifiers on the registered go label. "fix" is the declared MATH format-compliance branch; rows that contain no compliance term are invariant to it. The 3-feature row is the only significant pre-decision model that depends on neither the leak nor the fix.

Model	AUC (with fix)	95% CI	no fix	covers 0.5
Registered 12-feat (v1, leaked)	0.755	$[0.500, 0.929]$	0.755	yes
Pre-decision 10-feat	0.844	$[0.733, 0.943]$	**0.663**	no
L2-CV pre-decision	0.875	$[0.763, 0.960]$	0.722	no
3-feat (invariant)	**0.767**	$[\mathbf{0.646}, \mathbf{0.879}]$	**0.767**	no
No pass@8	0.837	$[0.721, 0.930]$	0.675	no
Semantic (MiniLM)	0.512	$[0.255, 0.746]$	0.512	yes
Metadata (4 scalars)	0.309	$[0.051, 0.675]$	0.309	yes

Median wall $(\text{S2}+\text{S3})/(\text{S4}+\text{S5}) = 0.17$ (IQR $[0.11, 0.36]$); including S1, 0.53 (IQR $[0.35, 1.19]$). pass@8 is 1600 generations per task and is not required for the gate (no-pass@8 row).

1.4 Primary Result

In this regime the probe is a *ranker with measured budget value*, not a stoplight worth building operations around. Spending $K{=}20$ full slots by Δ_{pilot} harvests 72% of all positive gain, near the oracle 74%. The registered binary endpoint is negative and stays negative after the post-decision feature is removed (0.663); only a parsimonious three-feature pre-decision model is significant (0.767). Even that is almost useless as a *decision*, because a perfect gate saves 2.4% of positive-gain mass relative to always forging. The 87% base rate, 23/61 zero-base format unlocks, and McNemar's two significant negatives are the same fact: no-go is scarce. Instruction text does not replace the probe (semantic AUC 0.51). We do not reopen BIRD/APPS, do not move the registered go threshold, and do not treat sensitivity (b) as evidence.

2 System Design and Meta-Analysis

2.1 Agent and Harness

We ran the study with two coupled pieces: a frozen Python harness that executes one task from package load through a sealed `metrics.json`, and an LLM coding agent that packed those tasks, drove the remote GPU worker, diagnosed failures, ran the pre-registered analysis, and drafted this manuscript. The experimental model inside the harness is Qwen2.5-1.5B-Instruct at a pinned revision; the coding agent's weights are omitted in this report.

The harness is a single-task state machine S0–S6: load and hash-check the package; greedy-evaluate the unadapted base; train a 10-example LoRA probe; evaluate the probe; train the full LoRA; evaluate the full adapter; seal metrics (systems block included). Resume is journal-boundary only: a killed full-training stage restarts at S4 with S0–S3 kept. The agent never edits the protocol file. It may add scripts, isolate a failed run directory, or stop and write a report.

Figure 4 is the loop that produced the sealed $n{=}61$ table: a task factory writes frozen packages; a serial queue feeds the S0–S6 runner; gates mark PARTIAL or `over_budget` without inventing labels; incremental then full harvest brings artifacts back; a pre-registered script seals the analysis.

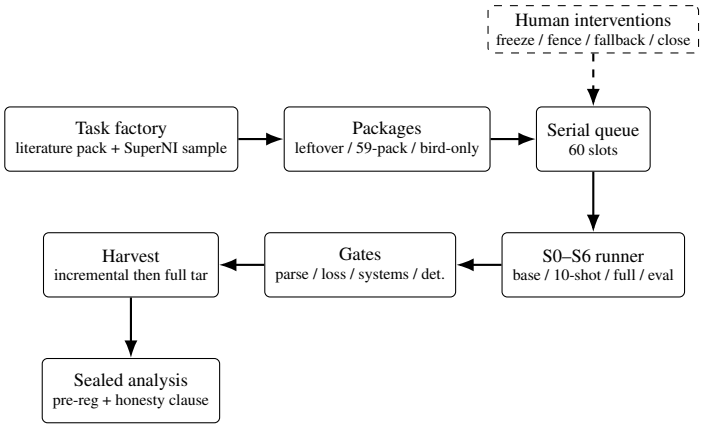

Figure 4: Factory → frozen packages → serial queue → S0–S6 runner → gates → harvest → sealed analysis. Dashed box: human intervention points (freeze, fence, fallback, close).

2.2 Tools

Compute was one 48GB-class GPU worker and one serial queue. The worker ran the harness under a terminal multiplexer; the agent supervised it with scheduled probes of `journal.jsonl`, `STATUS`, and `metrics.json`, because a remote job does not wake the session when it ends. Local tools were git on a single mainline (small commits, no force-push), an append-only ledger, pytest, and SHA-256 manifests. Execution-based verifiers (GSM8K, WinoGrande, Spider, later MATH/MBPP/DROP/TyDiQA/ARC/HellaSwag/PIQA and a SuperNI exact-match family) expose a uniform `verify(output, reference) -> {pass, parsed, note}`; unparseable is `parsed is None`, never a silent wrong. Each verifier has mutation tests: gold must pass, a known-wrong answer must fail, format variants of a correct answer must pass, garbage must unparse. GSM8K accepts `#### 1,000` as 1000 and rejects a hash without a number; WinoGrande will not treat the bare English article "a" as choice A; Spider compares execution multisets and treats a 30 s timeout as fail with the exception type in `note`. The factory pins source revisions at pack time (dataset SHAs in the package `task.json`) and writes a per-package `MANIFEST.sha256`. SuperNI clones were frozen at `55a36563`... before ids were committed; English+size filter left 890 surviving tasks, then the seed-20260822 draw of 50 with a `Source[0]` cap of 3. Code execution is a subprocess with a 10s cap, no network, and a 2 GiB virtual-memory limit.

2.3 Research Loop

The loop was not an open-ended search over architectures. It was discover (pack a frozen task), pilot (10-shot LoRA), decide (record Δ_{pilot} as a signal, not a gate), train (full LoRA), validate (greedy pass@1), then stop. The scientific hypothesis — whether that probe classifies go $\Leftrightarrow \Delta_{\text{full}} > 0$ — was written down before the remaining production runs.

Protocol freeze came first. After three smoke runs the operator stopped a proposed `protocol_v3`: the pilot 10×100 reaching loss $\sim 10^{-5}$ is recitation, and recitation is the probe, not a bug. `protocol_v2.yaml` was never edited after that freeze; any later exception is a run-level flag, not a silent yaml change.

Sample-frame correction came next. An earlier count treated 46 retestable *pool rows* as 46 tasks. Deduped literature-layer tasks are 13 (3 already sealed, 10 new) plus 50 SuperNI tasks drawn by a pre-registered English-and-size filter (seed 20260822, `Source[0]`≤ 3). Intended n=63. A second GPU worker was cancelled; the 60 new slots ran as one serial list. BIRD was packed later as its own package; the 59-task package did not wait. Mini-Dev was never used.

The serial queue started 2026-08-22T09:34Z and drained 60/60 at 2026-08-24T00:22Z on that one GPU (39 h wall). Of those 60 slots, 55 sealed `STATUS=ok` on the 59-task package; the rest were PARTIAL or `over_budget` (TyDiQA and task419 S4 OOM, ARC-Easy unparseable at 5% with metrics, BIRD S0 field error, APPS 3 h S1-only). Three earlier smoke runs (GSM8K, WinoGrande, Spider) already counted as production data under the freeze. A per-task 3 h wall marks `over_budget`; gates run after S6 and do not block the next task. Determinism uses a 200-row rerun on list items 1 and every 10th, otherwise 30 rows. Incremental harvest pulled a 10-directory tar after every tenth GPU completion (adapters omitted). Failed directories are isolated, not deleted. A four-task rerun wave (declared before any AUC) then ran 2026-08-25T06:19Z–13:11Z: BIRD dual-sha checker, TyDiQA and SuperNI task419 with a disclosed 1×16 OOM fallback (effective batch still 16), APPS at an 8 h ops fence. Analysis was sealed the same day at n=61 under the honesty clause. No AUC was computed until that script ran.

2.4 Human Interventions

Human interventions were discrete, dated, and written into the ledger before the next GPU step. We list the ones that changed what the sealed table contains, including the agent's own misreports.

False-negative connectivity. The agent reported that the GPU worker was unreachable. A follow-up over an already-open multiplexed session returned `ok`. The cause was the agent's tool layer refusing the remote-shell call, not a host or authentication failure. Production started after that correction, not after a re-provision. That is an agent error in the log, not a cluster outage.

Procedure slip, then a STOP. Gate 1 (WinoGrande, then Spider) was started before Gate 0's five-eye table had been accepted. The operator halted Spider at the S4 boundary (SIGTERM; S0–S3 kept), kept the WinoGrande directory, and only then passed Gate 0. Spider later resumed from that journal boundary and sealed. The STOP did what a comment in chat cannot: it prevented a protocol decision from being made on a moving GPU.

Format overwrite, not EOS. GSM8K returned $\Delta_{\text{full}} = -0.230$ with unparseable $0/200$ and a deterministic greedy rerun. A truncation/EOS reading does not fit those numbers. Base format-compliance was #### 58 / last-number 142 / none 0; after the probe and after full training the slice was #### 200/200. The operator's Gate 0 ruling — format overwrite with reasoning collapse — is the one we keep. The derived `format_compliance` field was backfilled into metrics and did not bump the protocol version.

Schema lag on BIRD, failed closed. The first BIRD slot died in S0 on `KeyError: sha256`: the checker wanted a single `source.sha256`; the package stored `train_zip_sha256` and `dev_zip_sha256`. No metrics were written. The directory was isolated. The checker was aligned to the dual-sha fields; the bird-only package was not rebuilt. A later slot passed S0, then died of CUDA OOM at the first probe backward pass (S1 pass@1 0.085 / pass@8 0.23; still no `metrics.json`). We did not invent a SQL label to close the queue.

Wall as an ops fence, not a measurement. APPS hit the 3 h cap in S1 with no metrics (`over_budget`). The 8 h rerun was declared before any APPS number existed. It finished S1 at base pass@1 0.0 / pass@8 0.005 and died of CUDA OOM at full-training step 22/495 — still no Δ_{full}. The floor is a missing-negative footnote, not a row.

OOM fallback, two tasks only. TyDiQA and SuperNI task419 died of CUDA OOM in S4 on the first pass. A run-level 1×16 fallback (effective batch 16, yaml untouched, flag in STATUS and metrics) was declared before those reruns, and only those two task ids may carry the flag. Both resealed: $\Delta_{\text{full}} + 0.530$ / $+0.505$. ARC-Easy stayed PARTIAL (unparseable $10/200 = 5\%$, metrics present) and entered the main sample; sensitivity (a) drops it.

Campaign close. BIRD and APPS stay out: `excluded for logistics, before any label existed`. The go threshold, the protocol file, and the honesty clause were not reopened after seeing AUC.

What the operator always owns. The following decisions never moved to the agent, even when the agent drafted the patch: freeze or not freeze the protocol file; keep or replace the 10×100 probe; the honesty clause and the go threshold; whether Mini-Dev may substitute for BIRD; the exact exclude sentence and the rule that it may be used only before a label exists; whether a PARTIAL with metrics (ARC-Easy) enters the sample; whether the pod may be killed; the disclosure boxes in this PDF. The agent owns packing, the unattended queue, harvest SHA checks, five-eye tables, and writing `PARTIAL` instead of a guessed Δ. When those two columns mixed — the SSH misreport, Gate 1 started early — the log names the mix rather than rewriting it.

2.5 Verification

Four checks sit under every sealed row, a fifth sits under the paper's numbers, and a sixth — independent re-implementation — is the one that actually caught something.

Gates (automatic, do not block the queue): base unparseable $< 5\%$; pilot loss must fall; the systems block must be complete (torch, transformers, CUDA, driver, GPU name, base revision, seeds, `dry_run=false`); a greedy rerun of the unadapted base must match per-id `pass` (full 200 on list items 1 and every 10th; otherwise 30 rows; seed 20260820). Failure writes `STATUS=PARTIAL`. Smoke three-task five-eye was all green (GSM8K/WinoGrande unparseable $0/200$, Spider $1/200 = 0.5\%$; rerun mismatch 0).

Packages carry `MANIFEST.sha256`. Incremental harvest tars omit adapters and were SHA-checked against the worker; the full archive (adapters and isolated dirs included) has sha256 `71309f7c...f4028f7d`. Figures in Part 1 are redrawn from a frozen plot payload, not from a second analysis.

The check that caught a mis-attribution. The revision's headline — "after removing the post-decision feature the gate is significant" — survived code review and survived rerunning the authors' own script. It did not survive an *independent* re-implementation of the LOTO pipeline: that reproduced AUC 0.663, not 0.844. A cell-by-cell diff of the two 61×10 design matrices found exactly one disagreement, a format-compliance value for MATH, and a decomposition showed that removing leakage moves the AUC *down* while that single cell moves it up by 0.182. The lesson is narrow and reusable: re-running the pipeline verifies determinism, not correctness, and neither a frozen protocol nor a written contract detects a feature-definition change that both the code and its author believe is a bug fix. Only a second implementation compared at the level of the design matrix does. The finding is disclosed in `ANALYSIS_PREREG_V2` as a late declaration and the paper's claim was moved onto a model invariant to it.

The analysis contract — go iff $\Delta_{\text{full}} > 0$, task-level LOTO logistic, Spearman on the two deltas, text-only control, bootstrap 95% CI, "AUC non-significant = publish negative" — was committed before the remaining 60 GPU slots. The coding agent drafted this Part 2 from that ledger and those reports; a human author then walked every claim against the ledger and the sealed reports before signing the disclosure attestations.

2.6 Significance of result

The qualifying result is a 61-row, execution-based ground-truth table that did not exist before the harness ran, plus a pre-registered reading of it: forging is the default (53/61), the 10-shot probe ranks gain, and the binary gate is not established. Small-model post-training spends real GPU time on tasks that look like GSM8K (negative transfer) and on tasks that look unimportant until a probe is actually run (description-only AUC below chance). A ranker that is honest about not being a stoplight is usable for budget allocation; a non-significant AUC hidden by post-hoc features would not be. The process claim, for this report, is narrower: a frozen protocol, a pre-registered honesty clause, and fail-closed logistics were enough to publish that negative gate rather than wait for more no-go tasks.

2.7 Interpretability

Protocol files, task packages, and run directories are append-only versioned (`protocol_v2`, `tasks_v3` leftover, `tasks_v4` 59-pack, `tasks_v5` bird-only). Old versions remain on disk; a run directory names the protocol and package it consumed. The ledger is append-only. Journal lines are stage-boundary events; Spider is the existence proof that resume does not replay S0–S3. Harvest is two-tier by design: incremental tars for monitoring (no adapters), one full tar for audit. Isolated failed dirs live inside that tar. A sealed run directory is meant to be readable without the agent: `task.json` and the package manifest, `journal.jsonl` stage boundaries, `eval_*_greedy.jsonl` for every labelled slice, `STATUS`, and `metrics.json` with the systems block (torch / transformers / CUDA / driver / GPU name / pinned base revision / seeds / `dry_run=false`). If that bundle is missing, the row is not a label. That is why BIRD and APPS are logistics excludes rather than no-go.

2.8 Limitations

The harness measures one base model, one LoRA recipe, greedy pass@1, and a $\Delta_{\text{full}} > 0$ gate that is empirically easy (8 no-go out of 61). Cluster AUCs were pre-declared and are not estimable. BIRD and APPS produced no label; we do not read that as "this model cannot learn SQL or competition-style code." Campaign GPU-hours were not summed and are not invented here. We report only walls that sit in sealed reports: smoke-task S0$\rightarrow$S6 of GSM8K $\approx$ 2.16 h, WinoGrande $\approx$ 0.51 h, Spider $\approx$ 2.22 h effective (plus a killed half S4 $\approx$ 0.9 h not counted in the 2.22); a TDP-times-wall upper bound of $\approx$ 1.5 kWh on that three-task block, not a metered reading; serial production $\approx$ 39 h wall for 60 slots; Wave-1 TyDiQA $\approx$ 92 min and task419 $\approx$ 53 min inside the 3 h cap. The agent's connectivity misreport and the Gate 1-before-Gate 0 start are in the log because hiding them would make the STOP discipline look cleaner than it was.

3 Broader Impact

Guardrails earned their keep when they failed *closed*. The BIRD checker aborted on a missing `sha256` key rather than packing a substitute hash or writing a fake `metrics.json`; the later CUDA OOM likewise produced no label. GSM8K's negative Δ_{full} came with unparseable $0/200$ and a format-compliance shift from mixed `####`/last-number at base to `####` 200/200 after training — evidence against an EOS/truncation story, and the reason the operator could rule "format overwrite" instead of discarding a negative as a generation glitch. A rail that invents a number to keep a queue green is not a rail.

Pre-registration is what made the negative primary endpoint publishable. The honesty clause (bootstrap CI covering 0.5 equals a negative, to be printed) was committed before the remaining sixty GPU slots and before any AUC existed. After the seal, the tempting moves — add a feature, drop GSM8K, move the go threshold, reopen BIRD — were exactly the moves the contract forbids. With 8 no-go tasks the gate was always going to be under-powered; the clause prevented that fact from being negotiated away in the write-up.

STOP points were the human–agent interface, with a mixed record. The protocol-v3 STOP (keep 10×100; recitation is the probe) and the analysis STOP (no production before the SuperNI ids and the pre-reg were on the mainline) are the reason Part 1 has one recipe and one contract. The cost is real: Gate 1 started before Gate 0 was accepted, and the connectivity misreport looked like downtime until a human asked for a one-line multiplex check. A STOP that exists only as a chatbot apology does not stop a GPU.

Diagnosis was split on purpose. The agent is responsible for five-eye tables, SHA checks, journal resume, and for writing `PARTIAL` rather than a guessed Δ. The operator is responsible for mechanism rulings (format overwrite), for ops-fence vs measurement (APPS 3 h then 8 h), and for the exclude sentence that may be used only *before any label existed*. We would like the field to copy that split: freeze the recipe, pre-register the honesty clause, fail closed, and keep the operator on the one-way door out of the sample.

4 Disclosure Statement

Agent(s) / model(s) used The 61-row experimental table was produced by a frozen Python harness on a single serial GPU queue (protocol v2; base model Qwen2.5-1.5B-Instruct at the pinned revision in the protocol file). An LLM coding agent packed tasks, monitored the queue, diagnosed failures, executed the pre-registered analysis and the declared CPU-only revision, and drafted Parts 1–3 of this manuscript, including figures generated from the frozen plot payload. The revision's own attribution error was found by an independent re-implementation of the analysis and is disclosed as a late declaration in the revision contract; no experimental number was recomputed on new hardware. The coding agent's model identity is omitted in this report. No experimental number in Part 1 was computed outside the sealed analysis script. Human status of this manuscript: data, hashes, and analysis artifacts were produced under the harness and the pre-registration, and a human author has since line-checked this text against the sealed sources. The two attestations below are signed on that basis.

- ☒ A human author has curated this work and verified every claim, figure, methodology detail, and reference.
- ☒ A human author takes full accountability for this report as a human-authored, human-verified artifact.

References

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. PIQA: Reasoning about physical commonsense in natural language. *arXiv preprint arXiv:1911.11641*, 2019.

Jonathan H. Clark, Eunsol Choi, Michael Collins, Dan Garrette, Tom Kwiatkowski, Vitaly Nikolaev, and Jennimaria Palomaki. TyDi QA: A benchmark for information-seeking question answering in typologically diverse languages. *arXiv preprint arXiv:2003.05002*, 2020.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? try ARC, the AI2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018.

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

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. *arXiv preprint arXiv:1903.00161*, 2019.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the MATH dataset. *arXiv preprint arXiv:2103.03874*, 2021.

Long-Kai Huang, Ying Wei, Yu Rong, Qiang Yang, and Junzhou Huang. Frustratingly easy transferability estimation. *arXiv preprint arXiv:2106.09362*, 2021.

Kevin Jamieson and Ameet Talwalkar. Non-stochastic best arm identification and hyperparameter optimization. *arXiv preprint arXiv:1502.07943*, 2016.

Lisha Li, Kevin Jamieson, Giulia DeSalvo, Afshin Rostamizadeh, and Ameet Talwalkar. Hyperband: A novel bandit-based approach to hyperparameter optimization. *arXiv preprint arXiv:1603.06560*, 2018.

Cuong V. Nguyen, Tal Hassner, Matthias Seeger, and Cedric Archambeau. LEEP: A new measure to evaluate transferability of learned representations. *arXiv preprint arXiv:2002.12462*, 2020.

Qwen Team. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. WinoGrande: An adversarial winograd schema challenge at scale. *arXiv preprint arXiv:1907.10641*, 2019.

Kaichao You, Yong Liu, Jianmin Wang, and Mingsheng Long. LogME: Practical assessment of pre-trained models for transfer learning. *arXiv preprint arXiv:2102.11005*, 2021.

Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, Zilin Zhang, and Dragomir Radev. Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-SQL task. *arXiv preprint arXiv:1809.08887*, 2018.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*, 2019.