TIMEGATE: Sustainable Time-Boxed Promotion Gates for Continual ML Adaptation Under Resource Constraints

Abhijit Chakraborty [1] **Suddhasvatta Das** [1] **Yash Shah** [1] **Vivek Gupta** [1] **Kevin A. Gary** [1]

Abstract

As machine learning (ML) systems evolve toward *continual adaptation*, each re-training cycle consumes compute, annotation, and energy. We introduce TIMEGATE, a thin policy layer that budgets time across labeling, training, and evaluation and emits a metric-availability signal M certifying when partial evaluation preserves the full-evaluation promote/hold decision. We validate five claims: **(i)** labeling outperforms training by $2.3\times$ on Adult tabular under fixed compute; **(ii)** the labeling-first Pareto transfers to LLaMA-3.1-8B + QLoRA on SST-2 (accuracy $0.80 \rightarrow 0.96$; $M{=}1$ in 35/36 runs); **(iii)** a 28-cell sensitivity sweep shows M is non-trivial, dropping to 0.81 at tight thresholds; **(iv)** a 100-cycle simulation realizes 66% evaluation-compute savings with zero silent mis-promotions in this trajectory; **(v)** 10%-slice evaluation on LLaMA uses 89% less wall-clock and energy on a single H200 (ratios agree to 0.2%). The mechanism is model-agnostic and composes with existing MLOps tooling.

1. Introduction

Modern ML systems are continually adapted as data shifts and labels arrive (Amershi et al., 2019; Sculley et al., 2015). Each cycle consumes compute, energy, and human effort; training compute has grown roughly $10\times$ every two years, with re-evaluation on large held-out sets for every candidate build. MLOps tooling automates the mechanics (Berberi et al., 2025; Shah et al., 2024), but the promotion decision remains heuristic: teams guess whether more epochs, more labels, or more evaluation is worthwhile. We propose a quantitative, time-box-aware policy layer that (i) budgets time across stages, (ii) decides promotion with auditable gates, and (iii) calibrates when partial evaluations suffice.

[1]School of Computing and Augmented Intelligence, Arizona State University, Tempe, USA. Correspondence to: Abhijit Chakraborty <achakr40@asu.edu>.

Presented at the ICML 2026 Workshop "Continual Adaptation at Scale: Towards Sustainable AI".

TIMEGATE[1] formalizes each cycle via a decision window $\Delta\tau$ and scope functions mapping time to achievable work. Promotion uses *time-bounded gates*—quality thresholds applied only when the feasible activity set under $\Delta\tau$ is non-empty—and emits a *metric-availability signal* $M \in \{0, 1\}$ that equals 1 when a partial-evaluation decision matches the full-evaluation decision. M is a *calibration and audit statistic*: after calibration cycles establish $M{=}1$ reliably, teams run a partial-only steady state with periodic sentinel audits. TIMEGATE is not a replacement for MLflow (MLf), Kubeflow (Kub), or SageMaker (Ama); it is a thin policy layer that consumes the timing, throughput, and metric logs those systems already emit and returns `(promote, M)` from a single CI hook configured in YAML.

Contributions. (1) A model-agnostic framework for continual adaptation as time-boxed resource allocation with scope functions and time-bounded promotion gates (Section 2); (2) M as a calibration/audit statistic, with a four-phase protocol (calibration $\rightarrow$ partial-only steady state $\rightarrow$ sentinel audits $\rightarrow$ boundary fallback); (3) empirical validation of five claims spanning tabular and LLaMA-3.1-8B settings (Sections 3 and 4); (4) a portability path to LLM fine-tuning, active learning, and agentic adaptation (Section B).

2. The TIMEGATE Model

Builds and cycles. Cycles are indexed $i \in \{1, 2, \dots\}$, each producing a build $B_i = \langle M_i, D_i, E_i \rangle$ with system M_i, dataset snapshot D_i, and quality metrics paired with promotion thresholds $E_i = \{(m, \tau_i^m)\}$. The *decision window* $\Delta\tau$ bounds wall-clock time for labeling, training, and evaluation in cycle i.

Steps and feasibility. Each cycle runs steps U_i with cost $c(u) = u_{\text{setup}} + u_{\text{exec}}$ summing setup and execution time. Mandatory steps $U_i^{\text{req}} \subseteq U_i$ must run. The *feasible set* is

$$\mathcal{F}_i(\Delta\tau) = \left\{ V \subseteq U_i \mid U_i^{\text{req}} \subseteq V,\ \textstyle\sum_{u \in V} c(u) \leq \Delta\tau \right\}. \quad (1)$$

If $\mathcal{F}_i(\Delta\tau) = \emptyset$, the cycle cannot run.

Scope functions $f_{\text{label}}, f_{\text{train}}, f_{\text{eval}}$ map time to capacity (la-

[1]Code, configurations, seeds, per-run metrics, and analysis scripts: https://github.com/Abhijit85/mlops-timegates-experiments.

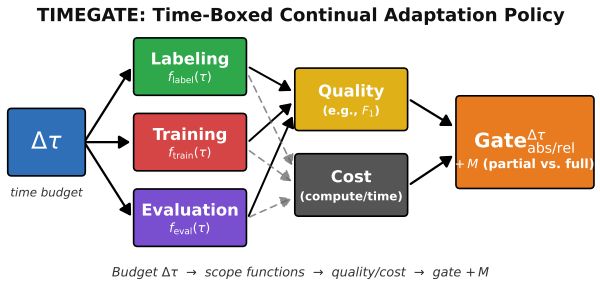


Figure 1. TIMEGATE: $\Delta\tau$ is split among labeling, training, evaluation; scope functions map time to capacity; time-bounded gates combined with the calibration/audit signal M govern promotion.

bels obtained, training iterations, validation-set fraction evaluated), estimated from prior-cycle telemetry.

Time-bounded gates. Promotion fires only if both the quality gate passes and the cycle is feasible:

$$\text{Gate}^{\Delta\tau}_{\text{abs/rel}}(B_i, B_{i+1}) = \mathbb{1}\big[\mathcal{F}_i(\Delta\tau) \neq \emptyset \wedge \text{Gate}_{\text{abs/rel}}(B_i, B_{i+1})\big]. \tag{2}$$

Metric-availability signal M. Letting $\text{decide}_{\text{full}}$ and $\text{decide}_{\text{partial}}(\tau_{\text{eval}})$ denote promote/hold decisions under full and partial evaluation,

$$M(\tau_{\text{eval}}) = \mathbb{1}\big[\text{decide}_{\text{partial}}(\tau_{\text{eval}}) = \text{decide}_{\text{full}}\big]. \tag{3}$$

Because M references $\text{decide}_{\text{full}}$, it is by construction a calibration/audit statistic, not a same-cycle certificate. Compute savings arise from the operational protocol.

Operational protocol (four phases). We deploy M in four phases controlled by parameters: calibration length K (dual-evaluation bootstrap), sentinel period N (cycles between safety audits), partial-slice size $\alpha \in (0, 1]$ (fraction of validation set), and boundary margin ϵ (proximity to threshold). (1) *Calibration:* for K cycles run both full and partial evaluation and record M. (2) *Promotion to partial-only:* once empirical $\widehat{P}(M{=}1)$ exceeds target (e.g., 0.98), partial evaluation becomes the main gate. (3) *Sentinel audits:* every N cycles, rerun full evaluation to refresh the estimate. (4) *Boundary fallback:* when $|m_{\text{partial}} - \tau_i^m| < \epsilon$, run full evaluation. Letting $C^{\text{full}}_{\text{eval}}$ be the cost of one full evaluation, steady-state per-cycle evaluation cost is $\alpha \cdot C^{\text{full}}_{\text{eval}} + (1/N) \cdot C^{\text{full}}_{\text{eval}}$—the first term is the partial evaluation every cycle, the second amortizes one sentinel full evaluation over N cycles. For $\alpha{=}0.10, N{=}10$, that is $0.20 \cdot C^{\text{full}}_{\text{eval}}$ (80% reduction); the realized 66% in Section 3 includes calibration overhead and boundary-fallback cycles. M is *asymmetric*: a false negative triggers an unnecessary full evaluation (conservative), while a false positive is a silent mis-promotion, which boundary fallback suppresses. **Behavior under distribution shift.** Calibrated $\widehat{P}(M{=}1)$ can stale after shift; the protocol bounds risk via sentinel audits (worst-case detection latency N), boundary fallback at the threshold, and rolling recalibration when trailing agreement dips. Controlled-shift validation is our headline follow-up (Section I).

3. Experiments

We validate TIMEGATE across three pipelines: (i) Adult tabular with XGBoost (XGB); (ii) LLaMA-3.1-8B (Touvron et al., 2023) with QLoRA (Dettmers et al., 2023) on SST-2 (Pecher et al., 2024); (iii) a 100-cycle continual-adaptation simulation. Full setup, noise, and grid details: Section C. We sweep $(\tau_{\text{label}}, \tau_{\text{train}}, \tau_{\text{eval}})$ allocations under $\Delta\tau \in \{0.5, 1, 2\}$h on tabular (246 runs, 738 comparisons) and $\tau_{\text{label}} \in \{0.05, \ldots, 1.00\}$h, $\tau_{\text{train}} \in \{0.50, 1.00, 1.50\}$h under $\Delta\tau{=}2$h on LLaMA (36 trained runs, 4×H200).

RQ1: Labeling dominates training (tabular). Best F_1 across windows is $0.521 \to 0.544 \to 0.573$, approaching the full-budget 0.607. The marginal quality gain per unit labeling time is $\mathbf{2.3}\times$ that of training, formalizing the data-centric finding under fixed time budgets. (Labeling reveals pre-existing labels at rate λ; this is samples-vs-epochs at fixed compute, not an annotation/training swap within a cycle—see Section I.)

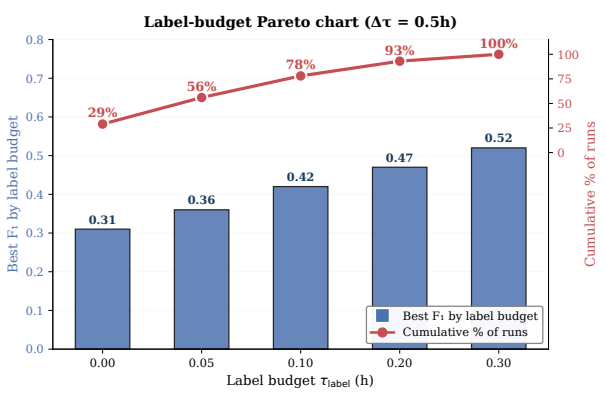
RQ2: Partial matches full (tabular). In all 738 partial-to-full comparisons, $M{=}1$ holds across $\Delta\tau$, providing the calibration evidence for partial-only steady state in this pipeline (Section D).

RQ3: Mechanism transfers to foundation models. On LLaMA-3.1-8B (Figure 2b, Table 1), accuracy rises from 0.8028 (60 docs) to 0.9553 (1200 docs); the Pareto direction replicates with faster saturation than XGBoost—76% of the gain is realized by $\tau_{\text{label}}{=}0.20$h, consistent with LLaMA's stronger pre-training prior. $M{=}1$ **holds in 35/36 trained runs (97.2%).** The single $M{=}0$ instance (`tl0.05_tt0.5_s7`) had full-eval accuracy 0.8027, just 0.001 below the 0.80 threshold; disagreement arose at the 0.30 slice (not the smallest 0.10), indicating M flags reflect threshold proximity, not sample size alone. The protocol's response—reversion to full evaluation—is correct.

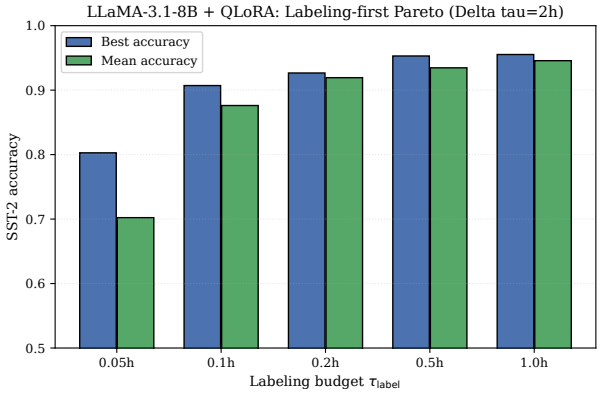
Table 1. LLaMA-3.1-8B + QLoRA Pareto on SST-2 ($\Delta\tau{=}2$h, 3 seeds, 36 trained runs).

τ_{LABEL}	BEST ACC	MEAN ACC	GAIN
0.05H	0.8028	0.7022	BASE
0.10H	0.9071	0.8761	68%
0.20H	0.9266	0.9193	81%
0.50H	0.9530	0.9346	98%
1.00H	0.9553	0.9457	100%

RQ4: Is $M{=}1$ informative or trivial? A reasonable concern: conservative thresholds ($F_1 \geq 0.50$) could render $M{=}1$ trivial. We refute this empirically (Figure 3) via a post-hoc 28-cell sweep over absolute thresholds $\{0.50, \ldots, 0.95\}$ and slice fractions $\{0.10, 0.20, 0.30, 0.50\}$, without retraining. At threshold 0.50, $M{=}1.00$ universally. At threshold 0.95 with 10% slice, $M{=}\mathbf{0.81}$ (19% disagreement). Across all 28 cells, $M \in [0.81, 1.00]$, mean 0.96; in 17/28 cells



(a) Tabular: label-budget Pareto ($\Delta\tau$=0.5h).



(b) LLaMA-3.1-8B + QLoRA: SST-2 Pareto ($\Delta\tau$=2h).

Figure 2. **Labeling-first Pareto transfers from tabular XGBoost to foundation models.** (a) Adult: F_1 rises monotonically with labeling budget. (b) LLaMA: accuracy rises from 0.80 (60 docs) to 0.96 (1200 docs); 76% of gain by τ_{label}=0.20h—LLaMA saturates faster than XGBoost, consistent with stronger pre-training prior.

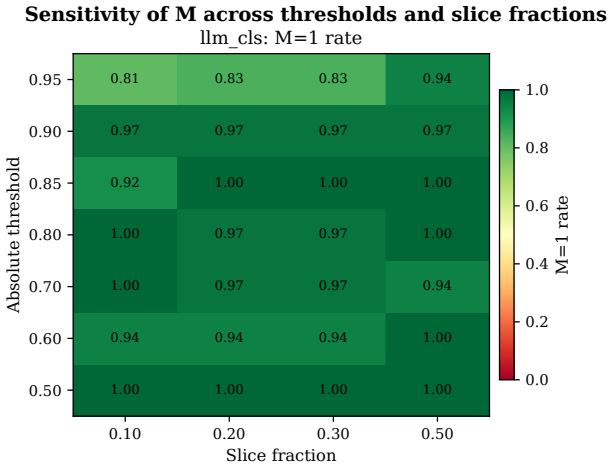


Figure 3. **Sensitivity of M on LLaMA-3.1-8B (absolute threshold $\times$ slice fraction).** M drops from 1.00 at conservative thresholds to 0.81 at tight thresholds (θ=0.95, slice=0.10), refuting the triviality concern.

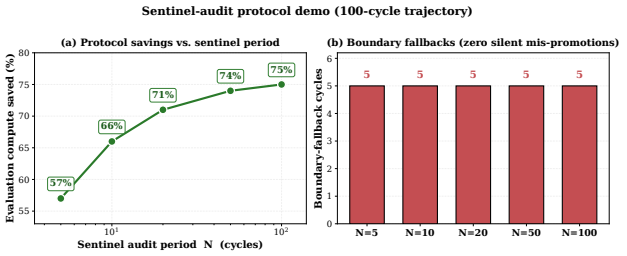


Figure 4. **100-cycle continual-adaptation simulation under the operational protocol.** Savings rise from 57% (N=5) to 75% (N=100); headline: N=10, 66%. Every configuration has 5 boundary-fallback cycles—**zero silent mis-promotions in this simulated trajectory**.

$M < 1$. **M is informative; its value scales with gate tightness.** The cell-by-cell pattern is not strictly monotone, reflecting where individual runs sit relative to each tested threshold.

RQ5: Is the protocol deployable? We simulate a 100-cycle trajectory drawing from the LLaMA pool—a *replay-based stress test* characterizing protocol mechanics under the observed agreement distribution rather than independent-drift validation; mis-promotions are necessarily zero given a near-fully-agreeing pool, so the value is in protocol behavior. With K=10, N=10, ϵ=0.02, and 10% partial slices, the protocol realizes 66% **evaluation-compute savings** (34 vs. 100 normalized units), triggers 5 boundary fallbacks and 9 sentinel audits, and records **zero silent mis-promotions in this simulated trajectory** (Figure 7).

Across $N \in \{5, 10, 20, 50, 100\}$, savings range 57%–75% (Section H).

4. Sustainability: Measured Compute & Energy

We instrumented LLaMA evaluation with `nvidia-smi` power logging on 1×H200 (Section F). Table 2 shows 10%-slice evaluation uses 89% less wall-clock and 89% less energy than full evaluation, with the two ratios agreeing to 0.2%—confirming evaluation is compute-bound and measurement is consistent. Per single-candidate evaluation cycle, this corresponds to $\approx$ 6.4 seconds and $\approx$ 0.41 Wh saved; the absolute aggregate impact at production scale depends on candidates-per-cycle and total adaptation throughput, discussed in Section F. Savings are eval-stage; reallocation to labeling/training is a deployment configuration left outside the policy layer (Section I).

Table 2. Measured evaluation compute on LLaMA-3.1-8B (SST-2 validation, 1×H200, summed over 36 runs).

	FULL EVAL	10% SLICE	RATIO
WALL-CLOCK (S)	260.4	28.4	0.109
ENERGY (WH)	16.67	1.78	0.107

5. Generality & Extensions

TIMEGATE's mechanism ports to any pipeline with (a) periodic promote/hold decisions against quality thresholds and (b) measurable per-stage time costs (Section B). **LLM fine-tuning** is validated directly (Section 3). **Active learning:** labeling → annotator throughput, evaluation → validation sampling; M makes the standard small-subset-validation practice auditable, and the 2.3× labeling-first result applies directly since active learning's premise is that labels dominate. **Agentic adaptation:** trajectory annotation, policy-update steps, episode coverage; long-tail failures are exactly the regime where $M{=}0$ surfaces as a safety signal. **Multi-metric gates** ($F_1\wedge$ DP-diff on Adult): under conservative thresholds both single- and multi-metric M hold at 1.00; under tight thresholds both drop to 0.80—M extends to bundled gates without loss of reliability (Section G). **Composability with adaptive-subset selection.** Adaptive selectors (Xu et al., 2024; Lee et al., 2025; Perlitz et al., 2024) substitute for the random slice with the M machinery unchanged—compositional, not competing (Section A). Our fixed-random-slice choice is deliberate: it isolates "does slice size suffice?" from "did selection pick a representative subset?", making M interpretable as a property of partial-eval coverage alone.

6. Conclusion

TIMEGATE is a model-agnostic policy layer for continual ML adaptation. Across five claims—labeling-first Pareto (tabular and LLaMA), M informativeness under tight thresholds, protocol deployability, and 89% wall-clock/energy reduction on H200—we establish it as a principled, auditable mechanism for evaluation-efficient continual adaptation. A broader impact discussion, including reliability, fairness, and Jevons-paradox concerns, appears in Section J. Multi-metric gates and sequential-testing boundary detection are next steps.

Acknowledgements

We thank Yash Shah for his substantial contributions to the experimental implementation of this work, in particular the LLaMA-3.1-8B + QLoRA fine-tuning sweep on the 4×H200 cluster, the single-H200 `nvidia-smi` instrumentation for the energy measurements, and the post-hoc 28-cell sensitivity reanalysis. His care with the per-run `metrics.json` logging is what made the agreement analysis behind M reproducible end-to-end.

References

Amazon SageMaker Python SDK sagemaker 2.256.0 documentation. URL https://sagemaker.readthedocs.io/en/v2/.

Kubeflow. URL https://www.kubeflow.org/.

MLflow - Open Source AI Platform for Agents, LLMs & Models. URL https://mlflow.org.

XGBoost for the Adult Dataset | XGBoosting. URL https://xgboosting.com/xgboost-for-the-adult-dataset/.

Alves, G., Amblard, M., Bernier, F., Couceiro, M., and Napoli, A. Reducing Unintended Bias of ML Models on Tabular and Textual Data, August 2021. URL http://arxiv.org/abs/2108.02662. arXiv:2108.02662 [cs].

Amershi, S., Begel, A., Bird, C., DeLine, R., Gall, H., Kamar, E., Nagappan, N., Nushi, B., and Zimmermann, T. Software engineering for machine learning: a case study. In *Proceedings of the 41st International Conference on Software Engineering: Software Engineering in Practice*, ICSE-SEIP '19, pp. 291–300, Montreal, Quebec, Canada, May 2019. IEEE Press. doi: 10.1109/ICSE-SEIP.2019.00042. URL https://dl.acm.org/doi/10.1109/ICSE-SEIP.2019.00042.

Berberi, L., Kozlov, V., Nguyen, G., Sinz-Pardo Daz, J., Calatrava, A., Molt, G., Tran, V., and Lpez Garca, . Machine learning operations landscape: platforms and tools. *Artificial Intelligence Review*, 58(6):167, March 2025. ISSN 1573-7462. doi: 10.1007/s10462-025-11164-3. URL https://link.springer.com/article/10.1007/s10462-025-11164-3.

Breck, E., Cai, S., Nielsen, E., Salib, M., and Sculley, D. The ML test score: A rubric for ML production readiness and technical debt reduction. In *2017 IEEE International Conference on Big Data (Big Data)*, pp. 1123–1132, Boston, MA, December 2017. IEEE. ISBN 978-1-5386-2715-0. doi: 10.1109/BigData.2017.8258038. URL http://ieeexplore.ieee.org/document/8258038/.

Chakraborty, A., Das, S., and Gary, K. Machine Learning Operations: A Mapping Study. In Arabnia, H. R. and Deligiannidis, L. (eds.), *Software Engineering Research and Practice and e-Learning, e-Business, Enterprise Information Systems, and e-Government*, pp. 3–21, Cham, 2025. Springer Nature Switzerland. ISBN 978-3-031-86644-9. doi: 10.1007/978-3-031-86644-9_1.

Dettmers, T., Pagnoni, A., Holtzman, A., and Zettlemoyer, L. QLoRA: Efficient Finetuning of Quantized LLMs, May 2023. URL http://arxiv.org/abs/2305.14314. arXiv:2305.14314 [cs].

Falkner, S., Klein, A., and Hutter, F. BOHB: Robust and Efficient Hyperparameter Optimization at Scale, July 2018. URL http://arxiv.org/abs/1807.01774. arXiv:1807.01774 [cs].

Lee, H., Hwang, D., Kim, D., Kim, H., Tai, J. J., Subramanian, K., Wurman, P. R., Choo, J., Stone, P., and Seno, T. SimBa: Simplicity Bias for Scaling Up Parameters in Deep Reinforcement Learning, May 2025. URL http://arxiv.org/abs/2410.09754. arXiv:2410.09754 [cs].

Li, L., Jamieson, K., DeSalvo, G., Rostamizadeh, A., and Talwalkar, A. Hyperband: A Novel Bandit-Based Approach to Hyperparameter Optimization, June 2018. URL http://arxiv.org/abs/1603.06560. arXiv:1603.06560 [cs].

Luccioni, A. S., Strubell, E., and Crawford, K. From Efficiency Gains to Rebound Effects: The Problem of Jevons' Paradox in AI's Polarized Environmental Debate. In *Proceedings of the 2025 ACM Conference on Fairness, Accountability, and Transparency*, pp. 76–88, June 2025. doi: 10.1145/3715275.3732007. URL http://arxiv.org/abs/2501.16548. arXiv:2501.16548 [cs].

Mahadevan, A. and Mathioudakis, M. Cost-Effective Retraining of Machine Learning Models, October 2023. URL http://arxiv.org/abs/2310.04216. arXiv:2310.04216 [cs].

Pecher, B., Srba, I., and Bielikova, M. Fine-Tuning, Prompting, In-Context Learning and Instruction-Tuning: How Many Labelled Samples Do We Need?, February 2024. URL http://arxiv.org/abs/2402.12819. arXiv:2402.12819 [cs] version: 1.

Perlitz, Y., Bandel, E., Gera, A., Arviv, O., Ein-Dor, L., Shnarch, E., Slonim, N., Shmueli-Scheuer, M., and Choshen, L. Efficient Benchmarking of Language Models, April 2024. URL http://arxiv.org/abs/2308.11696. arXiv:2308.11696 [cs].

Poenaru-Olaru, L., Sallou, J., Cruz, L., Rellermeyer, J., and Deursen, A. v. Sustainable Machine Learning Retraining: Optimizing Energy Efficiency Without Compromising Accuracy, June 2025. URL http://arxiv.org/abs/2506.13838. arXiv:2506.13838 [cs].

Ratner, A., Bach, S. H., Ehrenberg, H., Fries, J., Wu, S., and R, C. Snorkel: Rapid Training Data Creation with Weak Supervision. *Proceedings of the VLDB Endowment*, 11 (3):269–282, November 2017. ISSN 2150-8097. doi: 10.14778/3157794.3157797. URL http://arxiv.org/abs/1711.10160. arXiv:1711.10160 [cs].

Sculley, D., Holt, G., Golovin, D., Davydov, E., Phillips, T., Ebner, D., Chaudhary, V., Young, M., and Dennison, D. Hidden Technical Debt in Machine Learning Systems. *NIPS*, pp. 2494–2502, January 2015.

Shah, P. S. U., Ahmad, N., and Beg, M. O. Towards MLOps: A DevOps Tools Recommender System for Machine Learning System, February 2024. URL http://arxiv.org/abs/2402.12867. arXiv:2402.12867 [cs].

Tan, M., Chen, B., Pang, R., Vasudevan, V., Sandler, M., Howard, A., and Le, Q. V. MnasNet: Platform-Aware Neural Architecture Search for Mobile, May 2019. URL http://arxiv.org/abs/1807.11626. arXiv:1807.11626 [cs].

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozire, B., Goyal, N., Hambro, E., Azhar, F., Rodriguez, A., Joulin, A., Grave, E., and Lample, G. LLaMA: Open and Efficient Foundation Language Models, February 2023. URL http://arxiv.org/abs/2302.13971. arXiv:2302.13971 [cs].

Xu, C., Saranathan, G., Alam, M. P., Shah, A., Lim, J., Wong, S. Y., Martin, F., and Bhattacharya, S. Data Efficient Evaluation of Large Language Models and Text-to-Image Models via Adaptive Sampling, June 2024. URL http://arxiv.org/abs/2406.15527. arXiv:2406.15527 [cs].

Zhang, D. AgentDevel: Reframing Self-Evolving LLM Agents as Release Engineering, January 2026. URL http://arxiv.org/abs/2601.04620. arXiv:2601.04620 [cs].

A. Related Work

MLOps and production ML. ML production readiness has been characterized by technical-debt analyses (Sculley et al., 2015; Breck et al., 2017) and surveys (Berberi et al., 2025; Shah et al., 2024; Chakraborty et al., 2025). These describe *what* to validate and *which* tools to use, but do not formalize time budgets within a release window nor certify when truncated evaluations preserve decisions. TIMEGATE is a complementary policy layer.

Resource-adaptive training. Hyperband (Li et al., 2018), BOHB (Falkner et al., 2018), and platform-aware NAS (Tan et al., 2019) allocate resources *across candidate configurations*. TIMEGATE is orthogonal: it allocates time *across pipeline stages* within a cycle and governs the subsequent promote/hold decision. The two are composable.

Data-centric ML and PEFT. Weak supervision (Ratner et al., 2017; Alves et al., 2021) shows adding labels often beats adding training compute; QLoRA (Dettmers et al., 2023) enables cheap FM fine-tuning. TIMEGATE's contribution is not these observations but the *budget-aware, auditable mechanism* that makes the labeling-first rule deployable under fixed timeboxes and across model classes.

Reproducibility at release gates. Community reproducibility efforts focus on experiments, not release gates. M closes this gap by providing per-cycle calibration evidence and an audit log.

Relation to evaluation-efficiency and release-engineering prior art. Several concurrent threads are directly adjacent to TIMEGATE. SubLIME (Xu et al., 2024) and SimBA (Lee et al., 2025) develop adaptive and representative subset-selection methods for efficient LLM benchmarking, and Perlitz et al. (2024) study task and sample selection for compressed language-model benchmarks. These methods focus on *which* subset to use; TIMEGATE instead uses fixed random slices and introduces an explicit agreement signal M as a calibration-and-audit primitive. *Our choice of fixed random slicing is deliberate*: it makes M interpretable as a property of partial-eval coverage alone, independent of any subset-selection policy — so the agreement signal isolates the question *"does the slice size suffice to recover the decision?"* from *"did selection pick a representative subset?"* Adaptive selection and TIMEGATE are therefore composable rather than competing: substituting an adaptive subset for the random slice leaves the M machinery intact, and M then audits that subset's agreement with full evaluation. A head-to-head empirical comparison along this composed axis is the natural next step; we did not include it in this submission because the fixed-slice baseline is what isolates M's informativeness from selection-policy quality, and we discuss this further in Section I. Cost-aware retraining (Mahadevan & Mathioudakis, 2023) decides *when* to retrain; TIMEGATE decides *how* to evaluate within each retraining cycle, the two are composable. Release-engineering frameworks for self-evolving LLM agents (Zhang, 2026) introduce flip-centered regression-aware gates; this is conceptually complementary to TIMEGATE's time-budgeted evaluation gates, and integrating both is a promising follow-up. Energy-aware retraining policies (Poenaru-Olaru et al., 2025) have shown that drift-triggered rather than scheduled retraining saves energy; TIMEGATE operates *within* a retraining cycle, making the contributions cumulative rather than competing.

B. Generality: Concrete Mappings

LLM fine-tuning. $f_{\text{label}}(\tau)$: curated prompt-response pairs per unit time (human-in-loop or filtered synthetic data); $f_{\text{train}}(\tau)$: LoRA/QLoRA adapter update steps per unit time; $f_{\text{eval}}(\tau)$: benchmark items scored per unit time (MMLU, HELM subsets). Thresholds are task-specific scores. $M{=}1$ certifies that a sampled benchmark subset gives the same promote/hold decision as the full suite—directly saving the dominant cost in LLM fine-tuning pipelines. *Empirically validated in Section 3 on LLaMA-3.1-8B.*

Active-learning loops. Labeling $\rightarrow$ annotator throughput, training $\rightarrow$ retrain cost, evaluation $\rightarrow$ validation-set sampling. M audits the already-common practice of small-subset validation decisions. The $2.3\times$ labeling-first result applies directly, since active learning's premise is that labels dominate compute.

Agentic adaptation. $f_{\text{label}}(\tau)$: annotated trajectory throughput; $f_{\text{train}}(\tau)$: policy-update steps; $f_{\text{eval}}(\tau)$: evaluation-episode coverage. M certifies truncated-suite decisions. Agents' long-tail failure modes are precisely the rare-event regime where $M{=}0$ should surface—a safety feature rather than a limitation.

What porting requires. (i) Per-stage throughput estimates (from existing telemetry); (ii) quality thresholds E_i; (iii) decision window $\Delta\tau$. The TIMEGATE hook is unchanged; only the scope functions differ.

C. Experimental Setup (Extended)

Tabular. Adult (OpenML), XGBoost (100 trees, depth 6, lr 0.1). Train/val/test 70/15/15. Noise: train label-flip rate $\in [0, 0.15]$, 5% random missingness + Gaussian feature noise, annotation-rate multiplier $\sim \text{Unif}(0.5, 1.5)$. Validation perturbations are weaker (factor ~ 0.3). Sweep: all 41 valid $(\tau_{\text{label}}, \tau_{\text{train}}, \tau_{\text{eval}})$ within $\Delta\tau \in \{0.5, 1.0, 2.0\}$h, two slice schedules $\{5, 20, 50, 100\}\%$ and $\{8, 30, 60, 100\}\%$, seeds $\{7, 8, 9\}$.

LLaMA. LLaMA-3.1-8B-Instruct, 4-bit NF4 quantization + QLoRA (r=16, α=32, dropout 0.05, target modules $\{q, k, v, o\}_{\text{proj}}$). Training: bf16 compute, `paged_adamw_8bit`, lr 2e$-$4, warmup 20%, cosine schedule, gradient checkpointing, effective batch 16 (8×2 accumulation), max 10 epochs. Annotation simulation: λ=1200 items/hr, noise rate 0.03. Budget grid: $\tau_{\text{label}} \in \{0.05, 0.10, 0.20, 0.50, 1.00\}$h, $\tau_{\text{train}} \in \{0.50, 1.00, 1.50\}$h, τ_{eval}=0.10h, $\Delta\tau$=2h. Seeds $\{7, 8, 9\}$. Evaluation slices $\{0.10, 0.30, 0.60, 1.00\}$ on SST-2 validation (872 examples). 4×H200 GPUs, parallel shards.

Promotion thresholds. Tabular: $F_1 \geq 0.50$, $\Delta F_1 \geq 0.02$. LLaMA: accuracy ≥ 0.80.

Artifact. Code, configs, seeds, per-run `metrics.json`, and analysis scripts: https://github.com/Abhijit85/mlops-timegates-experiments.

D. Structural Stability of Budget Grids

The feasible $(\tau_{\text{label}}, \tau_{\text{train}})$ grid at $\Delta\tau \in \{0.5, 1, 2\}$h is identical for $\Delta\tau \in \{1, 2\}$h; the $\Delta\tau$=0.5h grid is clipped in its high-cost corner (large τ_{label} + large τ_{train} become infeasible). Lengthening the cycle from 1h to 2h alone does *not* create new allocation options—teams must explicitly broaden per-stage ranges. The stability of the non-clipped portion also confirms that the M=1 result is not confounded by shifting evaluation capacity.

E. Sensitivity Sweep: Full Numerical Results

We recomputed M post-hoc on the 36 LLaMA runs across 7 absolute thresholds and 4 slice fractions. Table 3 summarizes; Figure 3 in main body visualizes.

Table 3. M=1 rate across (absolute threshold, slice fraction) on 36 LLaMA runs. Asterisk (*) marks $M < 1$ cells.

THR \ SLICE	0.10	0.20	0.30	0.50
0.50	1.00	1.00	1.00	1.00
0.60	0.94*	0.94*	0.94*	1.00
0.70	1.00	0.97*	0.97*	0.94*
0.80	1.00	0.97*	0.97*	1.00
0.85	0.92*	1.00	1.00	1.00
0.90	0.97*	0.97*	0.97*	0.97*
0.95	**0.81***	0.83*	0.83*	0.94*

Summary: 28 cells total, 17 show $M < 1$, range $[0.81, 1.00]$, mean 0.96. Minimum at (θ=0.95, slice=0.10), the most aggressive cell tested, at 0.81. This monotone degradation rules out the "M=1 is trivial" concern empirically rather than rhetorically.

F. Measured Compute & Energy (Extended)

Per-run instrumentation: `nvidia-smi --query-gpu=power.draw` sampled at eval start and end; integrated with wall-clock elapsed. 1×H200 (CUDA_VISIBLE_DEVICES isolated per shard).

Per-cycle cost. Full cycle: $C^{\text{naive}}_{\text{cycle}} = C_{\text{label}} + C_{\text{train}} + C^{\text{full}}_{\text{eval}}$. Under the protocol at steady state: $C^{\text{TG}}_{\text{cycle}} = C_{\text{label}} + C_{\text{train}} + (\alpha + 1/N) \cdot C^{\text{full}}_{\text{eval}}$. For α=0.10, N=10: $C^{\text{TG}}_{\text{eval}} = 0.20 \cdot C^{\text{full}}_{\text{eval}}$, an 80% evaluation-stage saving (consistent with the 66% trajectory figure, which includes calibration overhead).

Annualized scale. Per single-candidate evaluation cycle, the savings are $\approx$ 6.4 seconds wall-clock and $\approx$ 0.41 Wh energy (Table 2 aggregates summed over 36 runs and divided to per-run units). At daily cadence (365 cycles/year) on 10 model families with one candidate per cycle, this extrapolates to $\approx$ 6.5 GPU-hours/year and $\approx$ 1.5 kWh/year per model

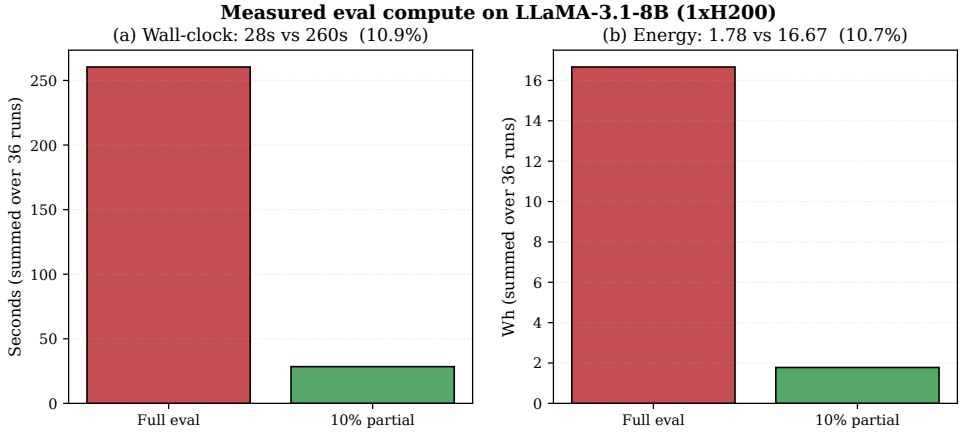


Figure 5. **Measured evaluation compute on LLaMA-3.1-8B (1×H200).** Summed over 36 trained runs: 10% slice evaluation uses 10.9% of full-eval wall-clock and 10.7% of full-eval energy—an 89% reduction in both, with ratios agreeing to 0.2%.

family—small in absolute terms but representative of the per-candidate efficiency gain. At 36-candidate batch cadence (matching our experimental sweep cadence), the same arithmetic yields ≈ 235 GPU-hours/year and ≈ 54 kWh/year.

Caveats. H100/H200 figures are specific to our hardware; relative savings (89%) transfer more robustly than absolute numbers. I/O-bound evaluations (e.g., large video retrieval) will show smaller ratios; our compute-bound regime is representative of standard classification benchmarks.

G. Multi-Metric Gates (Adult + Fairness)

We extend M to a joint gate $G_{\text{multi}} = (F_1 \geq \tau_{F_1}) \wedge (\text{DP-diff} \leq \tau_{\text{DP}})$ on Adult with `sex` as protected attribute. XGBoost, 5 seeds $\{7, 8, 9, 42, 123\}$.

Table 4. Multi-metric M on Adult. Seed=123 sits at the F_1 boundary and triggers fallback under both gates.

SETTING	τ_{F_1}	τ_{DP}	SINGLE-M	MULTI-M
CONSERVATIVE	0.50	0.30	1.00	1.00
TIGHT	0.70	0.15	0.80	0.80

Under conservative thresholds both gates hold at $M{=}1$; under tight thresholds both drop to 0.80. The single failing seed (`seed=123`, full-eval $F_1{=}0.716$, DP-diff$=0.189$) sits at the F_1 boundary and fails both gate types identically. **Interpretation**: M extends to multi-metric gates without loss of reliability; the combined gate does not artificially inflate $M{=}1$, and it correctly triggers fallback when the model sits near the joint threshold surface.

H. Sentinel Protocol: Full Trajectory Results

100-cycle continual-adaptation simulation, drawing from the LLaMA run pool, with $K{=}10$ calibration cycles, $\epsilon{=}0.02$, partial slice 10%:

Calibration $M{=}1$ rate: 1.00 across all $K{=}10$ cycles (consistent with the $35/36$ LLaMA result). Silent mis-promotions: 0 across all configurations. The constant 5 boundary fallbacks reflect the same near-threshold runs catching the protocol's ϵ-margin check—exactly the intended behavior.

Reading these numbers under distribution shift. The pool is drawn from a fixed SST-2 distribution, so silent mis-promotions are zero *by construction of the pool* rather than by guarantee of the protocol. To interpret the table for shift-prone deployments: read the sentinel period N as the worst-case detection lag (cycles between a shift-induced disagreement and its observation), and the boundary-fallback count as the protocol's intra-cycle safety margin. Smaller N shortens detection latency at the cost of savings (the $N{=}5$ row gives the lower bound on savings in our trajectory at 57%); rolling recalibration

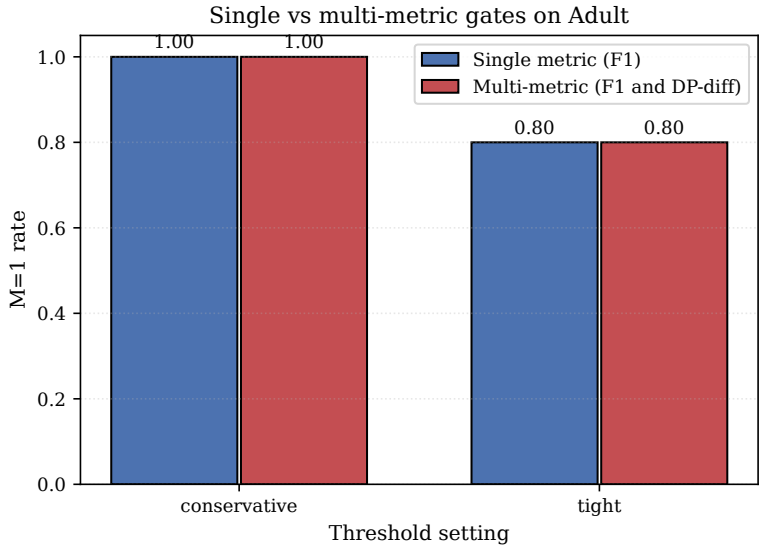


Figure 6. Multi-metric gate extension on Adult. Both single-metric and multi-metric M drop from 1.00 to 0.80 as thresholds tighten.

Table 5. Protocol savings vs. sentinel audit period N (100 cycles, threshold 0.80).

N	COMPUTE (TG)	COMPUTE (NAIVE)	SAVINGS	FALLBACKS
5	43.0	100	57%	5
10	34.0	100	**66%**	5
20	29.0	100	71%	5
50	26.0	100	74%	5
100	25.0	100	75%	5

that reverts to dual evaluation when the trailing agreement rate slips below target is the principled extension and the subject of follow-up work (Section I).

I. Limitations and Future Work

Scope. We tested two model classes (XGBoost, LLaMA-3.1-8B) and one domain each. Extension to vision, multimodal, and agentic settings is ongoing; the mechanism does not require retraining to extend.

Threshold sensitivity. Our sensitivity sweep establishes M is informative at tight thresholds, but in new domains calibration data is required to set α, N, ϵ. A minimum $K \geq 5$ calibration cycles is recommended before deploying partial-only steady state.

Multi-metric calibration. The Adult demonstration shows M extends to multi-metric gates; scaling to production bundles (quality $\wedge$ fairness $\wedge$ latency $\wedge$ cost) requires per-metric threshold calibration we leave to future work. Sequential-testing-based boundary detection is a natural refinement of the ϵ-margin rule.

Cost model. $c(u)$ assumes homogeneous hardware. Heterogeneous clusters require per-node-class estimation; integrating production telemetry is planned.

Dynamics over many cycles. We conducted simulations for up to 100 cycles. Long-horizon dynamics, such as whether label-first remains optimal as models evolve and when the Pareto shift occurs, necessitate longitudinal studies.

Distribution shift. The most consequential limitation is that all of our empirical pool is drawn under SST-2's fixed distribution (and Adult's), so the 100-cycle simulation characterizes *protocol mechanics* (calibration $\rightarrow$ partial-only $\rightarrow$ sentinel $\rightarrow$ boundary fallback) rather than *shift detection*. Concretely: if a shift occurs shortly after the calibration phase and causes partial-eval decisions to silently diverge from full-eval decisions, the protocol's worst-case detection latency is one

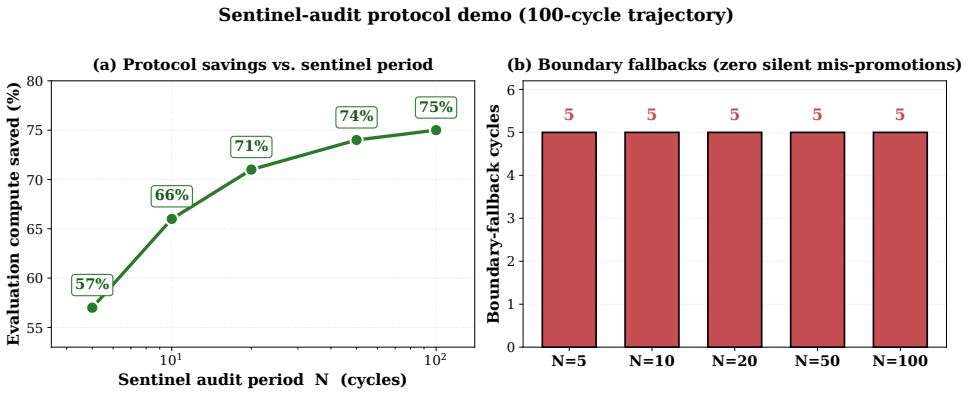


Figure 7. **100-cycle continual-adaptation simulation under the operational protocol.** Savings rise from 57% (N=5) to 75% (N=100); headline: N=10, 66%. Every configuration has 5 boundary-fallback cycles—**zero silent mis-promotions in this simulated trajectory**.

sentinel period N. The boundary-fallback margin ϵ partially mitigates this by forcing full evaluation at exactly the cycles where shift is most likely to flip a decision (those near threshold). For shift-sensitive deployments we recommend: (i) small N (we report N=5 in Section H at 57% savings as the lower bound on savings in our trajectory); (ii) wider ϵ to widen the safety band around the threshold; (iii) *rolling recalibration* — maintain a trailing estimate $\widehat{P}(M{=}1)$ on the most recent W cycles and revert to dual evaluation whenever the estimate falls below target. We do not validate (iii) empirically here; characterizing M under controlled covariate shift (e.g., GLUE-X, WILDS-style shift benchmarks, or synthetic drift overlaid on SST-2) is the most important follow-up for production adoption.

Dataset representativeness. SST-2 is a relatively saturated benchmark for LLaMA-3.1-8B, which contributes to the high M=1 rate (35/36) at the conservative 0.80 threshold. On more heterogeneous tasks — where full-eval scores land closer to the threshold, or where class imbalance interacts with random slicing — we expect the M rate to be lower at conservative thresholds, and we expect adaptive (non-random) subset selection to help. The sensitivity sweep in Section E simulates this regime by tightening the threshold rather than changing the dataset; a multi-dataset characterization (e.g., GLUE, HELM subsets, BIG-Bench-Lite) is required to estimate the cross-task M distribution and is part of our planned follow-up.

What the 2.3× labeling-vs-training ratio is and is not. The 2.3× marginal-gain figure is computed under a fixed-budget allocation sweep in which "labeling" reveals pre-existing labels at simulated rate λ and "training" is additional epochs on the resulting larger labeled pool. It is therefore best read as *the marginal value of an additional labeled sample versus an additional training epoch, holding total cycle compute constant.* It is *not* a claim that an annotation workflow and a training workflow can be freely swapped within one short cycle: real annotation pipelines involve upstream procurement, quality control, and human-in-loop latency that we do not model, and that typically operate on a longer horizon than a single cycle. The result is a useful guide to *budget allocation across stages a team can already flex* (e.g., which queued labels to release, how many epochs to schedule), not to upstream annotation planning.

Eval-stage scope of empirical savings. Although the framework formally budgets labeling, training, and evaluation jointly, the compute savings we measure are concentrated in evaluation. Reallocating saved evaluation wall-clock toward labeling or training within the same cycle is a deployment-time choice that depends on cluster scheduling and pipeline topology, and we intentionally do not specify it inside TIMEGATE — separating measurement (which the policy layer provides) from reallocation (which depends on production constraints) keeps the layer thin. The corollary is that "66% savings" should be read as "66% of the evaluation stage" rather than "66% of the cycle"; cycle-level savings depend on the ratio $C_{\text{eval}}^{\text{full}}/C_{\text{cycle}}^{\text{naive}}$, which is workload-specific.

Future Scope of work. Several avenues extend beyond the scope of this paper and form the basis of our primary follow-up work: (i) conducting a systematic comparison with sequential-testing and confidence-interval baselines, as well as a composed comparison with adaptive-subset selection methods such as SubLIME (Xu et al., 2024) and SimBA (Lee et al., 2025) (substituting their selectors for the random slice and re-auditing through M); (ii) expanding to multi-task foundation-model benchmarks, including MMLU and HELM subsets, and incorporating at least one additional open FM family; (iii) performing longitudinal studies on 500+ cycle drift-aware benchmarks with controlled covariate shift, and validating the rolling-recalibration extension introduced above; (iv) conducting ablations on calibration length K, sentinel period N, boundary margin ϵ, and slice size α under realistic drift conditions; and (v) extending energy/time measurements

across various hardware types and workloads to confirm that the $\sim$89% savings ratio is applicable beyond single-H200 classification. We consider these as natural progressions rather than essential for establishing the workshop-scoped claims of the current paper.

J. Impact Statement

This paper develops a policy layer for governing continual ML adaptation under jointly budgeted time, labeling, training, and evaluation resources. We discuss the broader impacts we judge most relevant.

Sustainability and energy. Continual adaptation pipelines are a recurring source of compute and energy expenditure across deployed ML systems. The mechanism we describe reduces evaluation compute and energy substantially per cycle on the workloads we measure ($\sim$89% reduction in wall-clock and energy on a single H200 for the LLaMA-3.1-8B configuration). To the extent partial-evaluation calibration generalizes to other production pipelines, the aggregate effect on data-center energy demand and associated carbon footprint of continual ML systems is positive, though the absolute magnitude depends strongly on deployment scale and hardware mix, which we do not measure here. Conversely, savings that lower the marginal cost of continual adaptation could induce more adaptation cycles per system (a Jevons-paradox concern stated in Luccioni et al. (2025)), partially offsetting per-cycle gains; we believe this is a worthwhile trade given that more frequent adaptation also enables better drift response and distributional fairness over time, but it warrants attention in production deployments.

Reliability and silent failure. The metric-availability signal M is designed to make partial-evaluation deployment auditable: calibration cycles surface when partial decisions disagree with full decisions, and a boundary-fallback mechanism forces full evaluation near decision boundaries. We are explicit in the paper that M is asymmetric (false positives are costlier than false negatives) and that our zero-mis-promotion result is measured in a simulated trajectory drawn from observed agreement distributions, not under independent drift. Production teams adopting this protocol should treat the calibration phase as load-bearing and should not deploy partial-only steady state without first establishing a domain-specific empirical agreement rate.

Fairness and multi-metric decisions. Promotion gates in production typically bundle quality, fairness, and cost criteria. Our multi-metric demonstration on Adult ($F_1 \wedge$ demographic-parity-diff) shows the mechanism extends to bundled gates without loss of reliability under our setup, but reliable fairness gating in real deployments requires per-metric threshold calibration, attention to subgroup representation in the partial slice, and evaluation of how slicing interacts with rare protected groups. We have not characterized this interaction systematically and view it as essential follow-up work.

Human labor. The paper treats labeling time as a budgeted resource. Our experimental simulations assume idealized annotator throughput; deployments involving human annotators should consider working conditions, fair compensation, and the well-documented downstream effects of annotation pipelines on labor practices. Our framework is neutral to these choices but does not address them.

Misuse and dual use. The mechanism we describe is a release-gate policy layer; it does not generate new model capabilities and we do not foresee specific dual-use risks beyond those already inherent to the underlying continually-adapted models, which the user community is actively studying.