Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Auditable Release Assurance for Quantized LLM Deployment Configurations: Anytime-Valid Non-Inferiority and Executable Certificates

Myunglae Jo
Kyungpook National University
Donghyun Kim
Palusomni Co., Ltd.
Changhan Son
Palusomni Co., Ltd.
Minseong Park
Palusomni Co., Ltd.
Il-Hyuk Yu
Palusomni Co., Ltd.
Hyungkwang Lim
lim@palusomni.com

Palusomni Co., Ltd.

Research Article

Keywords: software quality assurance, machine-learning systems, quantized large language models, non-inferiority testing, anytime-valid inference, reproducible certification

Posted Date: November 11th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-10959650/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

Auditable Release Assurance for Quantized LLM Deployment Configurations: Anytime-Valid Non-Inferiority and Executable Certificates

Myunglae Jo[a], Donghyun Kim[b], Changhan Son[b], Minseong Park[b], Il-Hyuk Yu[b,c], Hyungkwang Lim[b,d*]

[a]Department of Physics, Kyungpook National University, Daegu, 41566, Republic of Korea.
[b]Palusomni Co., Ltd., Seoul, 06628, Republic of Korea.
[c]Xivilization Co., Ltd., Fukuoka, 810-0001, Japan.
[d]Advanced Institute of Science and Technology (AIST), Kyungpook National University, Daegu, 41566, Republic of Korea.

*Corresponding author(s). E-mail(s): lim@palusomni.com;

Abstract

Quantization changes not only model size but the deployed software configuration, yet release decisions are often based on a single benchmark gap. We present an auditable software-quality gate for deciding whether a quantized large language model (LLM) may replace a reference configuration under a declared non-inferiority margin ε and false-certification budget δ. The method uses paired item-level outcomes and supports two distinct targets: future requests sampled from a task distribution and a committed finite audit frame inspected without replacement. Fixed-sample and anytime-valid procedures control false certification under their stated assumptions, while precommitment, candidate-selection rules, and a lifetime risk ledger address optional stopping and repeated releases. Protocol 3.1 binds the decision to a versioned deployment profile and to the supplied task, evaluator, frame, selection, and ordered-record files; a verifier that can be run by an independent third party replays the reference implementation's bindings, selected candidate, inspection order, and sequential-evidence checks. We evaluate the procedure on twelve measured model–task–quantization populations spanning NVIDIA Jetson Orin Nano and Mac deployment configurations. For the proposed procedures, the maximum observed false-certification rate across the simulated null-region cells was 3.8% for a 5% budget. Representative 500-item audit frames reached retrospective stopping points between 167

and 423 items. Cross-configuration parsed-prediction disagreement for the same quantized artifact reached 11.8%, supporting configuration-specific rather than file-specific release evidence. The current certificate example is a retrospective replay; prospective assurance additionally requires external anchoring.

Keywords: software quality assurance, machine-learning systems, quantized large language models, non-inferiority testing, anytime-valid inference, reproducible certification

ORCID: Myunglae Jo, 0000-0002-7960-3933.

1 Introduction

Deploying a quantized large language model in place of a reference model is a software-quality release decision. The released object is not only a weight file: it is a complete deployment configuration comprising the weights and quantization format, inference engine and kernels, runtime libraries, hardware and driver, tokenizer and prompt template, generation controls, and evaluator. Any component can change observed quality. Existing evaluation practice often reports how much average benchmark quality changes, but a release process additionally needs a bounded, auditable gate that states what configuration may replace what reference, for which target, under what tolerance and error budget. This paper supplies that gate for paired outcome records.

A common release heuristic is to compare point estimates: run the benchmark once on both models and ship if the accuracy gap is small. Under partial or adaptive evaluation, this heuristic alone does not control false certification. With 500-item benchmarks, a 2–3 point observed gap is compatible with both true non-inferiority and true regression; conversely, a lucky draw can mask a real regression. Worse, teams iterate—if the gap looks marginal they run the benchmark again, or extend it, and stop when the numbers look acceptable. Naive repeated testing inflates the false-acceptance rate in ways point estimates cannot reveal.

We formalize the release decision as *non-inferiority certification*: the null hypothesis is that the quantized model is inferior beyond tolerance (degradation $\geq \varepsilon$), and certification requires rejecting this null at level δ. The claim is deliberately one-sided — a quantized variant that *outperforms* its reference is no reason to block a release — so non-inferiority, not two-sided equivalence, is the correct target. This direction—burden of proof on the compressed model—matches the deployment risk asymmetry and mirrors non-inferiority trials in pharmacology. Our contributions:

1. **Software-quality release criterion** (§3): certification of a *deployment configuration* (model × quantization × engine × hardware) against a reference, for either a declared R1 task distribution or a committed R2 finite audit frame, with tolerance ε and risk δ; including the paired-record data model that makes the decision statistically efficient.
2. **Anytime-valid statistical protocol** (§4): a paired non-inferiority release-assurance protocol for binary correctness outcomes under both i.i.d. task sampling

(R1) and uniformly randomized finite-benchmark inspection (R2), with finite-sample validity theorems for each, a terminal census rule that decides every finite frame, a bounded-score sequential extension sketched, and a one-sided upper confidence sequence by e-process inversion; judge-mediated preference metrics are left to future work.

3. **Executable assurance evidence** (§5): a signed protocol with exclusive local-seed and public two-stage modes. Stage 1 fixes the order rule, complete candidate profiles, actual evidence bindings, margin, budget, and risk allocation; in adaptive R2 holdout a deterministic signed selection record is fixed before fresh Stage-2 audit randomness and order finalization. The issuer/verifier CLI recomputes signatures, actual-file bindings, selection, committed order, the wealth process, and the confidence curve; candidate-selection and lifetime risk accounting (Proposition 7); and an interface to a separately budgeted drift detector.
4. **Empirical validation on measured deployment outcomes** (§7): we evaluate operating characteristics by resampling from degradation distributions *measured across Jetson Orin Nano and Mac deployment configurations* (llama.cpp, GGUF quantizations of Qwen3-1.7B/8B (Yang et al. 2025) and Llama-3.2 (Grattafiori et al. 2024) models on MMLU (Hendrycks et al. 2021) and GSM8K (Cobbe et al. 2021)), including boundary populations whose empirical-population degradation sits at the tolerance. The observed simulation rates agree with the stated error budgets, while formal validity comes from the theorems under their assumptions.
5. **Configuration-specific quality evidence** (§7.8): in a one-device factorial the observed precision-by-backend interaction was -0.2pp (95% CI $[-1.0, +0.4]$pp) — no large backend interaction was observed, though this is not an equivalence claim — whereas the same artifact changed materially across machine/build configurations (11.8% parsed-prediction disagreement in an archived outcome-record comparison using a rebuilt Mac configuration; the original historical Mac build was not preserved) — so certificates must bind to configurations, not model files.

We do not claim novelty for exact paired tests, non-inferiority testing, e-processes, confidence sequences, or quantization evaluation in isolation. The contribution is their end-to-end composition into a software-quality release workflow: a one-sided release criterion, explicit control of optional stopping and candidate selection, configuration and evidence binding, certificate-decision replay, and a re-certification lifecycle.

2 Related Work

Measuring quantization degradation.

Post-training quantization methods are typically validated by perplexity and a benchmark battery. Representative methods include GPTQ (Frantar et al. 2023), AWQ (Lin et al. 2024), and SmoothQuant (Xiao et al. 2023). Such summaries hide behavioral change: in our own measurements, quantizations that move aggregate accuracy by ≤ 1.2pp still produce correctness-status changes on 0.6–12% of items (§7), consistent with the flip-level analyses in the paired-testing literature (Kübler et al. 2026; Singh 2026). Our work takes per-item degradation as the object of interest and asks how to make a release decision about it with error control.

Software quality and lifecycle assurance.

Breck et al. (2017) operationalize production-readiness testing through an ML test-score rubric; Ashmore et al. (2021) frame assurance across the machine-learning lifecycle; and Siebert et al. (2022) provide a quality-model perspective for ML systems. In a complementary supply-chain setting, Torres-Arias et al. (2019) provide general signed integrity guarantees for software artifacts and steps. This paper does not replace those frameworks. It supplies a narrower executable release contract tying paired non-inferiority, selection and randomization conditions, deployment/evaluator/evidence bindings, and replayable decision evidence.

Statistical rigor in LLM evaluation.

A growing line of work brings confidence intervals, power analysis, and paired designs to LLM benchmarking. These works target *estimation* (how uncertain is this benchmark number) or *comparison* (is model A better than model B). We target *non-inferiority certification* (is A at most ε worse than B, at confidence $1-\delta$), which reverses the null hypothesis and changes both the test and the sample-size calculus. Individual components have close precedents — exact McNemar-style paired degradation tests, anytime-valid non-inferiority tests, and lifecycle protocols separating certification from drift detection each appear separately in recent literature — and we position our contribution as the end-to-end release protocol and its validation on measured edge populations, not as the first use of any single test.

Anytime-valid inference.

E-processes and confidence sequences permit optional stopping at arbitrary data-dependent times (Waudby-Smith and Ramdas 2024). Recent systems work applies related machinery to runtime admission control for compression decisions during serving, applied at the precision of per-request KV-cache entries (Wei and Liu 2026; Wei et al. 2026). Those systems certify *runtime numerical distortion* of individual requests against an exact computation. We certify the *static deployment configuration* against a reference model at the task level, pre-deployment, with the certificate as the release gate; the two are complementary layers of an assured serving stack.

Non-inferiority and equivalence testing.

One-sided non-inferiority tests and TOST are standard in biostatistics and have been applied to model comparison in classical ML (Schuirmann 1987; Wellek 2010).

Directly adjacent work.

Three concurrent lines are closest to ours and shape what we can claim. Singh (2026) audits compressed language models with paired declared-margin equivalence tests, discordance-based sample-size calculations, and per-item evidence release; its registered anytime-valid extension is not yet executed. Zhou et al. (2026) develop CELEUS, which constructs anytime-valid confidence intervals for LLM evaluation from e-processes and targets general evaluation rather than release gating. Kübler et al. (2026) apply exact one-sided McNemar degradation tests to paired per-sample LLM

outcomes. Individually, paired evaluation, declared margins, discordance accounting, and e-process LLM evaluation are therefore established; our claim is the *composition* — a release-gate protocol binding an anytime-valid non-inferiority decision to a full deployment configuration, validated on flip distributions measured on edge hardware, with replayable decision evidence and re-certification lifecycle — together with the configuration-specificity evidence that motivates binding certificates to configurations at all.

3 Problem Formalization

Definition 1 (Deployment configuration)
A deployment configuration is a tuple $C = (M, Q, E, S, H, I, G, V)$: model source, revision, and artifact identity M; numerical and quantization contract Q; inference-engine source/build E; kernel, library, runtime, and operating-system inventory S; hardware, driver, and backend H; tokenizer, context, and prompt template I; generation parameters G; and metric, parser, scorer, and harness identity V. Protocol 3.1 realizes this tuple as a closed, versioned deployment configuration profile (DCP-v1; JSON version string `DeploymentConfigurationProfileV1`).

Let b denote the reference configuration and q the candidate. For a task distribution $\mathcal{D}$ over items x and a quality functional m, define the per-item paired degradation

$$d(x) \;=\; m\big(b(x),\, x\big) \;-\; m\big(q(x),\, x\big),$$

and the task-level degradation $\Delta = \mathbb{E}_{x\sim\mathcal{D}}[d(x)]$.

Under R1, a certificate concerns the declared task distribution; under R2, it concerns only the committed finite audit frame. In both regimes, the statement is specific to the recorded deployment configuration. It does not bound behavior off-distribution, under adversarial inputs, outside the committed R2 frame, or after any component of the serving path changes.

This certificate establishes relative non-inferiority to an independently accepted reference configuration; it is not an absolute task-suitability or production-readiness certificate. The numerical margins used in this study illustrate the protocol; production margins must be justified by application-specific quality requirements.

Definition 2 ((ε, δ) non-inferiority certification) A procedure π mapping samples to $\{\textsc{certify}, \textsc{abstain}\}$ (ε, δ)-certifies q against b if $\Pr[\pi = \textsc{certify}] \le \delta$ whenever $\Delta \ge \varepsilon$.

The hypotheses are $H_0 : \Delta \ge \varepsilon$ (inferior beyond tolerance) versus $H_1 : \Delta < \varepsilon$ (non-inferior): certification requires evidence *against* H_0, placing the burden of proof on the compressed model. Note ABSTAIN is not a claim that q is worse; power (certifying truly non-inferior configurations) is an efficiency property, quantified empirically in §7.

Metric scope.

This paper develops and validates the protocol for *paired binary* metrics — $m \in \{0, 1\}$ per item (exact match, pass/fail), $d(x) \in \{-1, 0, +1\}$, with the discordant-pair structure driving efficiency. Bounded continuous scores $m \in [0, 1]$ (so $d(x) \in [-1, 1]$) are covered by the same e-process machinery, whose validity only requires bounded payoffs; we state this as an extension and do not evaluate it empirically. Judge-mediated preference metrics raise separate issues (judge noise and bias, tie handling, position-order dependence) that require their own formalization and experiments; they are out of scope here.

Why paired?

The variance of the paired estimator scales with the discordance rate $p_w + p_b$ (the probabilities of harmful and beneficial flips, formalized in §4), not with the base accuracy variance. On our measured populations discordance is 0.6–21% (§7); across our measured MMLU configurations (0.6–20% discordance) the paired difference variance $p_w + p_b - \Delta^2$ is 2.4–78× smaller than the unpaired two-sample variance $a_b(1 - a_b) + a_q(1 - a_q)$ computed on the same records, largest at the lowest discordance (12.5× for Qwen3-1.7B Q8, 4.3× for Q4), so at equal model-inference budget (each design scores both models n times) a normal-approximation power analysis needs proportionally fewer items.

4 Certification Protocol

4.1 Sampling regimes and validity statements

Two distinct estimands must be kept separate. *(R1) Superpopulation*: items are drawn i.i.d. from a task distribution $\mathcal{D}$ and the certified quantity is $\Delta = \mathbb{E}_{\mathcal{D}}[d(x)]$ — the degradation on future requests. *(R2) Fixed audit frame*: the target is the mean degradation $\bar{\Delta}_N$ over a fixed, enumerated, committed set of N items (the *audit frame*), inspected sequentially without replacement in a *pre-committed uniformly random order*. The certified estimand is exactly the frame mean: our case study commits a 500-item MMLU frame and a 300-item GSM8K frame, so its certificates cover those frames, not the full benchmarks and not a task distribution. To certify a full benchmark, commit its complete item list as the frame — early stopping then certifies the full-benchmark mean without exhausting it, which is the natural deployment of R2; to certify future traffic, use R1. Our Monte Carlo operating-characteristic validation covers both regimes: the finite-frame benchmark analysis uses R2 — matching how release evaluation is actually run (each benchmark item evaluated once, no duplicates) — while the R1 simulations model repeated i.i.d. audit streams; the example artifact retrospectively replays the R2 wealth calculation on a fixed legacy order and is not presented as an R2-valid certification event (§5). R1 remains the right regime for production audit streams, where items arrive i.i.d. from live traffic.

Theorem 1 (Fixed-sample validity, R1) *Let $d_1, \ldots, d_n$ be i.i.d. with $d_i \in \{-1, 0, 1\}$ and $\Delta = \mathbb{E}[d_i] \geq \varepsilon$. Let $n_w = \#\{d_i = 1\}$. Then the rule "certify iff $F_{\mathrm{Bin}(n,\varepsilon)}(n_w) \leq \delta$" satisfies* $\Pr[\textsc{certify}] \leq \delta$.

Proof $\Delta = p_w - p_b \geq \varepsilon$ implies $p_w \geq \varepsilon$, so n_w stochastically dominates $\mathrm{Bin}(n, \varepsilon)$; the probability that a $\mathrm{Bin}(n, p_w)$ variable falls at or below the δ-quantile of $\mathrm{Bin}(n, \varepsilon)$ is at most δ by dominance. □

Theorem 2 (Anytime validity, R1) *Let $d_1, d_2, \ldots$ be i.i.d. with $\Delta \geq \varepsilon$, let $X_t = \varepsilon - d_t \in [\varepsilon - 1, \varepsilon + 1]$, and let each λ in a finite grid satisfy $0 < \lambda < 1/(1 - \varepsilon)$. Then each $W_T(\lambda) = \prod_{t \leq T}(1 + \lambda X_t)$ is a nonnegative supermartingale with $W_0 = 1$ (since $\mathbb{E}[X_t \mid \mathcal{F}_{t-1}] = \varepsilon - \Delta \leq 0$ and the factors are positive), hence so is the mixture $\bar{W}_T$. By Ville's inequality (Ville 1939),* $\Pr[\exists T : \bar{W}_T \geq 1/\delta] \leq \delta$ *for* every *stopping rule.*

The conditional-mean requirement in Theorem 2 is what fails under without-replacement inspection of a fixed benchmark (R2): past observations change the conditional mean of the remaining items, and the i.i.d. product e-process is no longer a supermartingale for the finite-population mean. The repair is to bet against the *conditional* null mean rather than the marginal one (Waudby-Smith and Ramdas 2020). The theorem relies on two assumptions that the implementation must then instantiate or justify:

Assumption 1 (R2 randomization) Conditional on the committed finite frame, the inspection order is generated independently of all candidate outcomes and is an *exact uniform random permutation* of the N item identifiers.

For selection-aware use, let $\mathcal{F}_{\mathrm{sel}}$ be the sigma-field generated by Stage 1, the disjoint selection-frame outcomes, and the deterministic selected-candidate record. The selected index S must be $\mathcal{F}_{\mathrm{sel}}$-measurable, and conditional on $\mathcal{F}_{\mathrm{sel}}$ the audit permutation must remain uniform and independent of every candidate's fixed audit-frame outcome array. Protocol 3.1 enforces this condition for R2 holdout with $K \geq 2$ by fixing S before fresh Stage-2 randomness is supplied. Conditional on selection, the existing single-candidate Theorem 3 then applies to the fixed audit-frame population of S when its null is true. This is an application under an explicit conditional-randomization premise, not a new general post-selection theorem.

A pre-committed but non-uniform order does not suffice. For example, take $\varepsilon = 0$ and the two-item population $(+1, -1)$. If a deterministic ordering places the beneficial item first, then $m_1 = 0$, $X_1 = 1$, and $W_1 = 1 + \lambda_1 > 1$, so the required supermartingale inequality fails. Precommitment provides integrity and replayability; outcome-independence must instead be supplied by the randomization mechanism and its chronology.

Assumption 2 (Per-item stability) For every committed item and configuration, the parsed outcome is fixed independently of the item's position in the inspection order: each item is evaluated in an isolated request with fresh context and KV-cache state, and no state is carried across audit items.

Theorem 3 treats $(d_1, \ldots, d_N)$ as a fixed finite population permuted by the inspection order; Assumption 2 is what makes the population well-defined. Repeated evaluation of a configuration in the *same* order (§7.4) verifies re-execution determinism but not order invariance; order invariance is additionally verified directly on one configuration: re-evaluating all 500 MMLU items on the certificate's candidate configuration (Qwen3-1.7B Q8, Jetson CUDA) in an independently shuffled order reproduced the identical parsed outcome for every item (500/500), and the stateless evaluation semantics fixed in the harness (below) extend the assumption to the remaining configurations.

R2 theorem domain and construction.

The following statement assumes $N \geq 1$, $\varepsilon \in [0, 1]$, and $\delta \in (0, 1)$. The bet grid is finite and precommitted. Each bet is predictable with respect to the prior filtration, and every wealth factor is nonnegative. Mixture weights are nonnegative, sum to one, and are fixed before inspection; the mixture guarantee is the fixed-convex-mixture lift of the component supermartingales. These conditions are in addition to Assumptions 1 and 2.

Theorem 3 (Anytime validity, R2) *Under Assumptions 1 and 2 (without replacement), let $(d_1, \ldots, d_N)$ denote the inspected sequence of a finite population with mean $\bar{\Delta}_N \geq \varepsilon$, let $S_{t-1} = \sum_{s<t} d_s$, and define the conditional null-boundary mean $m_t = (N\varepsilon - S_{t-1})/(N-t+1)$ and payoff $X_t = m_t - d_t$. Under H_0 the true conditional mean of d_t given $\mathcal{F}_{t-1}$ is $(N\bar{\Delta}_N - S_{t-1})/(N-t+1) \geq m_t$, so $\mathbb{E}[X_t \mid \mathcal{F}_{t-1}] \leq 0$. With per-step bet $\lambda_t = \min(\lambda, \frac{1}{2}/(1-m_t))$ (which keeps every wealth factor positive since $X_t \geq m_t - 1$), each $W_T = \prod_{t \leq T}(1 + \lambda_t X_t)$ is a nonnegative supermartingale and Ville's inequality bounds the false-certification probability by δ under every stopping rule. Three deterministic boundary cases are handled exactly: if $m_t > 1$ the null is combinatorially impossible (certify, no error possible); if $m_t < -1$ the null is guaranteed (abstain); and at $m_t \geq 1$ any bet $\lambda_t \in [0, \lambda]$ preserves validity: if $m_t = 1$ and the null is true, every remaining item must equal $+1$, so the payoff $X_t = m_t - d_t$ is almost surely 0 (and $m_t > 1$ is impossible under the null) — nonnegativity of the payoff alone would not in general give the supermartingale property, but this structural fact does. The reference implementations take $\lambda_t = \lambda$ — the continuous limit of the positivity clip from below — which Lemma 5 requires for the wealth process to be continuous and monotone in the margin. All shipped grids and certificates use this convention.*

Corollary 4 (Terminal census rule) *If the wealth process has not crossed $1/\delta$ by $t = N$, the entire population has been observed and $\bar{\Delta}_N = S_N/N$ is known exactly; the terminal decision "certify iff $\bar{\Delta}_N < \varepsilon$" is then deterministic and adds zero error probability. The combined procedure (e-process crossing, else census at N) therefore retains the δ guarantee of Theorem 3 and decides* every *population by N: certification of all truly non-inferior finite populations is guaranteed by the census step, not by the betting process alone, whose wealth need not cross the threshold on small or low-signal frames.*

The R2 procedure (with the census rule of Corollary 4) has a property the i.i.d. version lacks: every truly non-inferior finite population is certified by $t = N$ — by wealth crossing when the signal is strong, by census otherwise — at the cost that boundary populations consume most of the frame (Figure 2c).

4.2 Fixed-sample test

For the binary family with n paired records $d_i \in \{-1, 0, 1\}$, write $n_w = \#\{d_i = 1\}$ (flips worse) and $n_b = \#\{d_i = -1\}$ (flips better), so $\hat{\Delta} = (n_w - n_b)/n$. Our fixed-sample baseline is the unconditional exact binomial test of Theorem 1: certify iff $F_{\mathrm{Bin}(n,\varepsilon)}(n_w) \le \delta$. Because it uses only the harmful-flip count, it is finite-sample valid and nonasymptotic — generally conservative because of discreteness — for every null distribution and every n, including the zero-variance case (with no observed flips it certifies only when $(1-\varepsilon)^n \le \delta$). The price of this robustness is that it ignores n_b: on high-discordance populations where p_w and p_b are both large but nearly cancel, it has little or no power even when $\Delta \ll \varepsilon$ (§7).

Sample-size planning.

Because the exact test rejects on the harmful-flip count alone, its power is governed by p_w under the alternative, not by the net degradation Δ_1: for any alternative with $p_{w,1} > \varepsilon$ (e.g. $p_w = 7\%$, $p_b = 5\%$, $\Delta = 2\% < \varepsilon = 3\%$) the rejection probability tends to *zero* as n grows, and at $p_{w,1} = \varepsilon$ it stays at or below the level δ — in neither case does power approach one, so the test is valid but inconsistent against high-discordance non-inferior configurations — the operational reason the e-process is the primary instrument. When the exact test is used, the design alternative must be stated as $p_{w,1} < \varepsilon$ and the exact power computed directly: the smallest n with

$$\Pr_{n_w \sim \mathrm{Bin}(n,\, p_{w,1})} \left[F_{\mathrm{Bin}(n,\varepsilon)}(n_w) \le \delta \right] \;\ge\; 1 - \beta.$$

For $\varepsilon = 3\%$, $\delta = 5\%$, $p_{w,1} = 1.5\%$, $\beta = 0.2$ this gives $n = 645$ (rejection cutoff $n_w \le 12$, null level 0.0496, power 0.823). Sample planning for the e-process is done by simulation on pilot flip distributions.

4.3 Anytime-valid sequential test

The fixed-n rule fails under optional stopping. Following the bounded-mean betting construction of Waudby-Smith and Ramdas (2024), under R1 we construct an e-process for H_0 by betting: for payoff $X_t = \varepsilon - d_t \in [\varepsilon - 1,\, \varepsilon + 1]$ (under H_0, $\mathbb{E}[X_t] = \varepsilon - \Delta \le 0$), define wealth

$$W_T(\lambda) \;=\; \prod_{t=1}^{T} \big(1 + \lambda X_t\big), \qquad \lambda \in \Lambda \subset \big(0,\, 1/(1-\varepsilon)\big),$$

and the mixture $\bar{W}_T = \frac{1}{|\Lambda|} \sum_\lambda W_T(\lambda)$. Under H_0, $\bar{W}_T$ is a nonnegative supermartingale with $\mathbb{E}[\bar{W}_T] \le 1$; by Ville's inequality $\Pr[\sup_T \bar{W}_T \ge 1/\delta] \le \delta$. The decision

rule

$$\text{certify at time } T \iff \bar{W}_T \geq 1/\delta$$

is therefore (ε, δ)-valid *simultaneously over all stopping rules*: the certifier may monitor continuously, pause, extend the sample, or stop at any data-dependent time. Under R2 the same construction applies with the conditional payoff and per-step clipping of Theorem 3. All experiments use the fixed grid $\Lambda = \{0.05, 0.1, 0.2, 0.4, 0.8\}$, every element of which is below the positivity bound $1/(1-\varepsilon)$ for all margins considered.

Two scope conditions matter in practice. First, the regime must match the estimand: the i.i.d. process certifies the superpopulation mean (R1); the without-replacement process of Theorem 3 certifies the finite benchmark mean (R2) and requires the item order to be a pre-committed uniform permutation (§5). Second, certifying *many* candidate configurations and releasing the best-looking one is a selection event that individual δ-guarantees do not cover; the certifier must allocate risk across candidates (e.g. δ/K or a lifetime error budget, §5). A symmetric e-process for H_1 provides a futility signal (stop collecting, abstain); it is advisory only — stopping on it makes no inferential claim of inferiority and does not affect the certification error budget.

4.4 Net degradation versus correctness-status discordance

Δ is a *net* quantity: our GSM8K measurements include a configuration (Llama-3.2-1B Q4_K_M) whose net degradation is 2.7pp while 21% of items change correctness in one direction or the other. A release gate on Δ alone would treat a configuration that silently rewrites a fifth of behavior as equivalent to one that changes under 1% of it. When behavioral stability matters (regression tests, downstream consumers keyed to specific outputs), the same statistical machinery can be applied to a second arm on the *discordance rate* $\rho = \Pr[d(x) \neq 0]$ (correctness-status discordance) — note this counts changes in correctness status only; raw-output disagreement $\Pr[b(x) \neq q(x)]$ is at least as large and typically larger (§7.8): certify $\rho < \eta$ at level δ_ρ using the same machinery with the binary indicator $\mathbf{1}[d_i \neq 0]$ (this is a standard one-sided proportion test; the e-process applies unchanged). When the release rule is a *conjunction* — ship only if *both* arms (and all tasks) certify — no budget splitting is needed: this is an intersection-union structure, and under the global null at least one component null is true, so the global false-release probability is at most the per-component δ. Splitting is required only for the distinct goal of *joint coverage* of all reported confidence sequences, or when selection occurs (releasing if *any* of several candidates certifies, §7.7). Thus conjunctive criteria can be analysed by the stated intersection–union argument. The current protocol-3.1 reference implementation issues and verifies the primary net-degradation arm; executable multi-arm and multi-task issuance is left for future implementation.

4.5 Choosing ε and δ

ε is a product decision (how much degradation the workflow tolerates); δ is a risk budget per release decision. We recommend reporting the full certificate (below) rather

than a bare pass/fail, so a downstream consumer whose (ε', δ') were fixed *independently of the certificate evidence* can recompute the decision from the ordered records; choosing δ' after seeing $1/W_T$ is risk shopping, exactly the failure mode the commitment exists to prevent, and any consumer-side recomputation at a looser δ' than the issuer's committed allocation is a consumer decision, not an issuer certificate. When multiple task distributions matter, certification is per-frame/per-distribution, and the budget rule follows the release logic of §7.7: a conjunctive gate (release only if *all* tasks certify) needs no splitting — each component runs at full δ by the intersection-union argument; splitting is required only for joint coverage of all reported confidence sequences or when any selection occurs.

5 Executable Quality-Assurance Certificates and Re-certification

A protocol-3.1 certificate is a signed JSON document binding: (i) the complete supplied DCP-v1 objects for b and q; (ii) actual-byte and semantic bindings for the task/evaluation manifest, evaluator source bundle, audit frame, and, for holdout, selection frame and selection evidence; (iii) protocol parameters (ε, δ), regime, Λ, candidate-selection policy, and release allocation; (iv) the ordered per-item records, n, $\hat{\Delta}$, wealth, confidence sequence, and stopping time; and (v) the decision and an explicit threat-model scope. Record order is part of the evidence: replay requires the d_t stream, not a set digest. The integrity of a supplied complete profile (signature and JCS digest) and its completeness under the versioned schema/semantic contract are separate checks; neither establishes that the physical deployment assertions are true. The supplied software is a verifier that can be run by an independent third party to replay the reference implementation's checks. The verifier is not a second, independently developed implementation of the statistical procedure. Superseded certificates retain their version and validity period rather than being deleted, so an auditor can reconstruct what was certified at any past release. Because validity is anytime-valid, an auditor can extend the evidence at any later time — a same-distribution follow-up audit issues a NEW certificate version (possibly superseding the old one) rather than mutating it; a past certification event is a historical fact about its distribution and period and is never statistically "cancelled" by later wealth decay.

Pre-evaluation commitment.

A signed, replayable record is necessary but not sufficient: an issuer who first evaluates the whole frame could sort favorable items to the front and publish the certifying prefix. Protocol 3.1 therefore commits the order mode explicitly. In `public_two_stage_v1`, signed Stage 1 fixes the complete candidate universe, deterministic selection rule, actual task, evaluator, and frame bindings, estimand, risk budget, shuffle, and public-randomness source/round, but sets the local seed to null. For adaptive R2 holdout with $K \geq 2$, the required order is

$$M_1 \;<\; A_{\text{sel}} \;<\; M_2 \;<\; C,$$

where `select` consumes the actual disjoint selection frame and evidence, recomputes exact mean degradation for every committed candidate, applies `min_mean_degradation_v1` with lexicographic configuration-digest tie-breaking, and signs A_{sel}. Only then may fresh public randomness be supplied to `finalize`. Stage 2 binds the SHA-256 of the actual signed M_1 and A_{sel} files and derives its seed from the length-prefixed domain tag, both digests, and the committed source/round plus the supplied randomness value. It signs the derived seed and permutation digest before audit-frame evaluation. Issue and verify enforce every binding and strict internal UTC inequality. Local order is categorically rejected for adaptive R2 holdout with $K \geq 2$.

In `local_seed_v1`, Stage 1 instead requires one safe integer seed, sets public source/round to null, and forbids every Stage-2 object; `finalize` also refuses this mode. Local mode supports integrity and replay, but an operator-chosen seed alone provides neither seed-shopping protection nor Assumption 1. The historical measured certificate is consequently retained only as a version-2 retrospective replay. In both Protocol 3.1 modes raw unvalidated Stage-2 data cannot enter order derivation: shared preflight must first produce an immutable validated order-binding value. Assumption 1 cannot be literally discharged by any finite-seed implementation — it is *computationally instantiated*: the committed shuffle must at least be exactly uniform *given* ideal stream bits. The specification therefore pins a Fisher–Yates shuffle whose index draws are obtained from a seed-keyed SHA-256 counter-mode stream by *rejection sampling* to exact uniform integers (unbiased by construction; plain modulo reduction is disallowed because it biases index selection); the reference implementation ships this construction (`fisher-yates-sha256ctr-rejection-v1`) and new manifests pin it by name in a signed `shuffle_algorithm` field. Since the physical randomness is expanded by a cryptographic stream, the deployed guarantee is computational, and the gap is quantifiable:

Remark 1 (Computational randomization guarantee) A 256-bit seed indexes at most 2^{256} permutations, whereas uniformity over all $N!$ orders of a 500-item frame would require $\log_2 500! \approx 3767$ bits; no deterministic expansion is literally uniform. Instead, let $\mathcal{D}$ be the distinguisher that maps a permutation to the certificate's crossing event (a polynomial-time function of the committed procedure). Under the pseudorandomness of the committed stream construction,

$$P_{\text{impl}}\big[\text{false certification}\big] \;\leq\; \delta \;+\; \text{Adv}_{\mathcal{D}}^{\text{PRG}},$$

where $\text{Adv}_{\mathcal{D}}^{\text{PRG}}$ is the distinguishing advantage against the stream (negligible under standard assumptions on SHA-256 in counter mode). Theorem 3 assumes ideal uniform randomization; the implementation computationally instantiates it, and the deployed guarantee is δ up to this advantage.

One provenance disclosure applies: the retrospective version-2 example predates this profile and replays under its recorded legacy shuffle (Python's Mersenne-Twister `random.shuffle`). It demonstrates historical issuer/verifier replay, not protocol-3.1 randomization or full-configuration binding.

The computational interpretation additionally requires candidate selection before fresh audit-order randomness wherever selection is adaptive, one-shot use of the committed randomness source, and no abort, retry, seed grinding, or conditioning on the resulting order. A prospective historical claim requires external anchoring before the named randomness event. These conditions do not imply physical execution or remote attestation.

The protocol relies on three external trust assumptions. First, signatures establish internal ordering, not calendar truth: an issuer can backdate its own objects. External anchoring of Stage 1 and the selection record before the named randomness round is required for cryptographic prospectivity; an independently retained RFC 3161 timestamp token is one possible anchor mechanism (Adams et al. 2001), but anchor receipts and beacon truth are outside the Protocol 3.1 verifier. Second, a signed candidate manifest proves integrity of the *supplied* universe only. It cannot prove that no hidden candidates or precommit screening existed, prevent fragmentation into another release family, or establish completeness without external governance. Third, actual-file hashes establish digital correspondence, not that the claimed hardware executed the evaluator. Physical execution, remote attestation, key trust, and external source truth remain outside scope. Cached reference outputs may precede Stage 1 because they do not reveal paired degradation without the candidate outputs; candidate and selection outcomes used by the committed procedure may not be used to choose the later audit order.

Issuer, verifier, and decision checks.

The reference implementation supports five transitions: commit, select when required, finalize public randomness, issue, and verify. At each applicable boundary the issuer checks schema and semantic completeness, exact risk allocation, deployment and evidence bindings, candidate-selection chronology, order consistency, and signatures before emitting an object. The verifier repeats those contract checks from the supplied files, recomputes the selected candidate and inspection order when applicable, and recomputes every statistical state through the recorded stopping point. A malformed or inconsistent producer input yields no output object; a failed verifier check yields an invalid decision rather than a partial pass.

Signing uses Ed25519 as specified by RFC 8032 (Josefsson and Liusvaara 2017). Signed object bytes use RFC 8785 JCS canonicalization (Rundgren et al. 2020). Protocol 3.1 stores decision-authoritative risk budgets as reduced, bounded numerator–denominator pairs; human-readable decimals are derived fields. This represents allocations such as $0.05/3 = 1/60$ exactly. Margins and the finite bet grid retain canonical decimal strings, while counts and local seeds are schema-bounded integers within the I-JSON safe range.

The theorem statements concern ideal arithmetic; simulations retain binary64 as non-authoritative diagnostics. Issuance and verification instead share a conservative decision engine. It propagates exact rational inputs, accumulates outward-rounded decimal wealth intervals, accepts a wealth crossing only after strict interval separation, and otherwise abstains numerically. R2 wealth, deterministic-feasibility, and census

Table 1 Responsibility classification by assurance layer. "Pass — executed," "Detected — executed," and "Reject — executed" report checks actually run by the deterministic responsibility driver in Online Resource 2. For the post-order-selection and inexact-allocation cases, "not checked" indicates that the isolated layer contains no corresponding validity check; the full-protocol rejection was executed.

Case	Statistics alone	File integrity alone	Full Protocol 3.1	Exact reason
Valid committed workflow	Pass — executed	Pass — executed	Pass — executed	Paired decision, bindings, order, allocation, and signatures all replay.
Evaluator-source mutation	Pass — executed	Detected — executed	Reject — executed	The paired stream is unchanged, but the evaluator hash differs from the signed evidence binding.
Post-order candidate selection	Not checked by this layer	Pass — executed	Reject — executed	Adaptive R2 holdout with $K \geq 2$ prohibits local-order mode fixed before selection.
Inexact risk allocation	Not checked by this layer	Pass — executed	Reject — executed	Signed decimal `0.016667` is not the required exact allocation $0.05/3 = 1/60$.

states pass through one end-of-prefix evaluator after each observed record. Mandatory actual-file bindings cover the task/evaluation manifest, evaluator source, audit frame, complete deployment profiles, and ordered records; a holdout additionally binds its selection frame and evidence. Single-candidate, Bonferroni, and holdout policies then impose their respective candidate and release-risk conditions. Online Resource 1 gives the schemas, command lines, exhaustive negative cases, protocol versions and compatibility, and numerical boundary details.

Full-scope Protocol 3.1 fixtures are synthetic mechanics tests. The measured Qwen3-1.7B Q8/MMLU example is a legacy retrospective replay: it crosses at ε=5% after 283 items, but it does not establish a prospectively anchored Protocol 3.1 execution. A certification crossing is permanent at its stopping time; later records cannot revoke the historical crossing. The reported bound is obtained by inversion of the monotone test family (Lemma 5) and is stored with conservative upward rounding; the example stores $\widetilde{U}_T = 0.049851$. Detailed compatibility and repair fixtures are catalogued in Online Resource 1.

The responsibility analysis is not a throughput, accuracy, or product-performance ablation. It shows why the layers must be composed. Outside all three layers are the truth of physical execution, the truth of external timestamps or public-randomness claims, and the absence of hidden candidate pre-screening.

Lemma 5 (Monotonicity of rejection in the margin) *Fix the observed prefix $d_{1:t}$ and a bet parameter $\lambda > 0$. In R1, each wealth factor $1 + \lambda(\mu - d_s)$ is nondecreasing in μ. In R2, with $m_s(\mu) = (N\mu - S_{s-1})/(N - s + 1)$ and bet $\lambda_s(\mu) = \min\{\lambda, 1/(2(1 - m_s(\mu)))\}$ for $m_s < 1$ and $\lambda_s(\mu) = \lambda$ for $m_s \geq 1$ (the continuous limit of the clip, whose bound $1/(2(1-m_s))$ diverges as $m_s \uparrow 1$), each factor $1 + \lambda_s(\mu)(m_s(\mu) - d_s)$ is nondecreasing and* continuous *in μ: where the clip is inactive (including $m_s \geq 1$) the derivative in m_s is $\lambda > 0$; where it is active the factor*

equals $1+(m_s-d_s)/(2(1-m_s))$*, with derivative* $(1-d_s)/(2(1-m_s)^2) \geq 0$*; the branches agree at both switch points, and* $m_s(\mu)$ *is increasing in* μ*. Positivity holds throughout: for* $m_s \geq 1$ *the payoff* $m_s - d_s \geq 1 - d_s \geq 0$ *cannot lose wealth. Hence every* $W_s(\mu)$*, and therefore the event* $\{\sup_{s\leq t} W_s(\mu) \geq 1/\delta\}$*, is monotone in* μ*, so the non-rejected set is an interval and its supremum is well-defined and computable by bisection.*

Corollary 6 (Upper confidence sequence by inversion, with census override) *Restrict the inversion domain to product margins* $\mu \in [0,1]$*: every* λ *in our grid (*$\max\Lambda = 0.8$*) then satisfies the positivity condition* $\lambda < 1/(1-\mu)$ *for all* $\mu \geq 0$*, so each* $W_t(\mu)$ *is a valid e-process on the whole domain. This is a* protocol requirement*, not merely a property of our grid: the reference validator rejects any committed grid containing* $\lambda \geq 1$*, so the same mixture composition remains admissible throughout the inversion domain. (On* $\mu < 0$ *the largest bets violate positivity; bounds are truncated at* 0 *rather than extended to negative margins, so improvements are reported as* $U_t = 0$*.) For* $t < N$ *(and all* t *in R1) define*

$$C_t(\delta) = \Big\{\mu \in [0,1] : \textit{no e-process or deterministic rejection of } H_0(\mu) \textit{ by } t\Big\},$$
$$U_t(\delta) = \sup C_t(\delta) \quad (\textit{with } \sup\emptyset := 0),$$

where deterministic rejection means the R2 boundary case $m_s(\mu) > 1$ *(the null combinatorially impossible), and set* $U_t(\delta) = \max\{0, \sup C_t(\delta)\}$ *with the convention* $\sup\emptyset = 0$ *(a strongly improving model may reject even* $H_0(0)$*, emptying* C_t*; the reported bound is then* 0*). At census (*$t = N$*, R2 only) the population mean is known exactly and a* terminal override *replaces the inverted set:* $U_N = \max\{0, \bar{\Delta}_N\}$*. Exactness here is a statement about* rational *arithmetic:* $\bar{\Delta}_N = S_N/N$ *with integer* S_N *need not have a terminating decimal expansion (e.g.* $1/300$*), so the census decision* $S_N < N\varepsilon$ *is evaluated in exact integer/rational arithmetic, the certificate stores the exact pair* (S_N, N) *alongside the six-decimal field, and the stored decimal is the conservative upward rounding* $\widetilde{U}_N = \lceil \max\{0, S_N/N\} \rceil_{10^{-6}} \geq U_N$ *— the same directional convention as the bisection bound. The guarantee is* upper-bound coverage*, not set membership — the domain restriction truncates improvements at zero, so for every* $\Delta \in [-1,1]$ *(including* $\Delta < 0$*),*

$$\Pr_{\Delta}\big[\forall t : \ \Delta \leq U_t(\delta)\big] \geq 1 - \delta,$$

which holds trivially when $\Delta \leq 0$ *and by the e-process/census argument otherwise; the census step adds no error event, and a confidence sequence need not be nested, only uniformly valid. Note the override is what delivers* $U_N = \max\{0, \bar{\Delta}_N\}$*; the inverted e-process set alone need not shrink to the census value (a two-item population with* $d = (0,0)$ *and* $\mu = 0.1$ *is never rejected by wealth crossing, yet the census decides it).*

Implementation: U_t *is computed by bisection (justified by Lemma 5) to tolerance* 10^{-5} *and rounded* upward *to the stored six-decimal precision, so the stored bound* $\widetilde{U}_t \geq U_t$ *never understates risk; decision and display precision are thereby separated.*

Margin shopping and confidence sequences.

A single-ε decision invites post-hoc margin selection (fail at 2%, fail at 3%, publish the 5% pass). Two defenses are built in: the issuer's margin is fixed in the commitment and cannot be edited without breaking two checks, and the certificate carries a one-sided upper confidence sequence $U_t(\delta)$ obtained by e-process inversion — valid uniformly over time — so a downstream consumer whose own margin ε' was fixed externally can decide by comparing $U_T < \varepsilon'$ without any reliance on the issuer's margin choice.

Re-certification triggers.

The certificate no longer applies to the changed serving configuration when its fingerprints no longer match the serving path (engine upgrade, kernel change, hardware migration). Our cross-configuration measurements (§7.8) show that identical GGUF artifacts produce materially different correctness-status discordance distributions across machine/build configurations. Under input drift (the deployed distribution departs from certification $\mathcal{D}$), the estimand itself changes; streaming drifted items into the certification wealth process would conflate two distributions. Drift is therefore handled by a separate, separately-budgeted detector on a fresh audit stream whose alarm marks the standing certificate *superseded / out-of-scope* (not revoked) and triggers fresh certification on the new distribution (§8).

6 Experimental Setup

Testbed.

Our edge configuration is an NVIDIA Jetson Orin Nano (8 GB, JetPack 6, CUDA) running llama.cpp (Gerganov et al. 2023–) (build pinned; ngl=99, batch 1). Reference configurations run the BF16/F16 GGUF export of each model, on the Jetson CUDA configuration where memory permits (Qwen3-1.7B and Llama-3.2-1B) and on a Mac (x86-64) CPU build of llama.cpp where they do not (Llama-3.2-3B F16 and Qwen3-8B BF16) — reference outputs are computed once and cached. Rows whose reference and candidate ran on different machines are marked in Table 3 and constitute cross-machine/build comparisons; Table 2 gives the row-level configuration map.

Table 2 Deployment configuration map.

Family	Reference config	Candidate config	Relationship
Qwen3-1.7B (all)	Jetson CUDA BF16	Jetson CUDA Q8/Q4	same machine/build
Llama-3.2-1B (all)	Jetson CUDA BF16	Jetson CUDA Q8/Q4	same machine/build
Llama-3.2-3B Q4/Q8	Mac CPU F16	Jetson CUDA	cross machine/build
Qwen3-8B Q4	Mac CPU BF16	Jetson CUDA	cross machine/build
Qwen3-8B Q8	Mac CPU BF16	Mac CPU Q8	same machine/build

Configuration fingerprints are recorded in the artifact configuration manifests to the extent preserved: measured artifact SHA-256, engine commit, architecture, backend, `ngl`, driver/JetPack, harness (prompt+parser) hash, and generation parameters for the Jetson configurations; the Mac reference build's engine commit and build flags were not preserved (its llama-cpp-python version and generation parameters are recorded), and model revision and tokenizer hashes are not separately recorded for any row. Result rows are keyed by (model family, quantization, task, backend); the join to configuration fingerprints goes through the configuration bundles in

`configurations.json` (which cover the Jetson rows; explicit per-row configuration IDs are not stored in the summary table). All decoding is greedy (temperature 0) with fixed seed. We treat greedy decoding on a fixed build as deterministic: repeated evaluations on the six configurations listed in §7.4 showed no parsed-answer variation, and configurations not repeated are covered by this assumption rather than direct measurement. Statistical randomness enters through R1 resampling and R2 permutation randomization, not through stochastic decoding. The harness enforces the stateless evaluation semantics of Assumption 2: every item is a separate HTTP completion request to a persistent `llama-server` with its full prompt supplied fresh (no conversation state or shared history across requests), the server's per-request context and KV cache are populated only from that request's prompt, requests are strictly sequential at batch size 1, and the harness source (hashed in the configuration manifests) contains no cross-item mutable state beyond the output file.

Models and quantizations.

Qwen3-8B, Qwen3-1.7B, Llama-3.2-3B-Instruct, Llama-3.2-1B-Instruct; each as BF16/F16 reference plus Q8_0 and Q4_K_M GGUF quantizations. This spans (i) a size axis crossing the on-device memory boundary, (ii) two model families with different tokenizers and instruction styles, and (iii) two commonly used llama.cpp quantization levels.

Tasks.

MMLU (500-item fixed random subset, 0-shot forced choice) and GSM8K (300-item fixed subset, 0-shot chain-of-thought, exact match on final answer). Subsets were sampled once and reused verbatim across all configurations, enabling item-level pairing. The audit frames are identified by retained item identifiers and SHA-256 digests of the source task files — sufficient to bind the R2 estimand when the authorized task files are supplied to the verifier. The publishable review artifact omits copied benchmark questions and answers. The original subset-sampling script and upstream row identifiers were not archived, so the frames are identifiable by their retained records and hashes, not re-derivable from the upstream datasets by seed.

Paired records.

For each configuration and item we record correctness, the *parsed* prediction, generation length, and wall time. GSM8K (v3 harness) records additionally store the server's stop reason, a truncation flag, and the *full* generated output; the archived MMLU and legacy runs (v2 harness) store a 200-character output head and no stop metadata. Records are joined on item id to form degradation samples $d_i = m_b(x_i) - m_q(x_i) \in \{-1, 0, 1\}$. The publishable review archive removes those free-text output fields while retaining item identifiers, parsed predictions, correctness, timing, and stop metadata; this sanitization does not alter any paired degradation value.

The Qwen3-1.7B GSM8K rows are truncation-dominated stress cases: 287–293 of 300 generations reached the 512-token limit, so they should not be read as representative task-performing deployments.

Table 3 Measured paired degradation across deployment configurations. $\hat{\Delta}$ = reference minus quantized accuracy; flips = items changing correctness (worse/better); discordance = fraction of items with any flip. Rows marked † pair a Mac (x86-64, CPU build) reference with a Jetson (aarch64, CUDA build) quantized run — a cross-machine/build pair; unmarked rows hold machine and build fixed (reference and quantized both on the Jetson CUDA configuration, or both on the Mac CPU configuration as for Qwen3-8B Q8_0). Rows are keyed by (model family, quantization, task, backend); machine/build fingerprints, where preserved, are in the artifact configuration manifests (per-configuration bundles, not per-row identifiers). Complete retained raw reference–candidate records exist for eight of the twelve populations. Four populations support downstream replay through authoritative derived flip arrays and summaries because historical Mac reference runs were not preserved. The package does not reconstruct every historical execution environment, and record replay is not physical-execution attestation. See §7.8.

Model	Quant	Task	n	acc_b	acc_q	$\hat{\Delta}$ (pp)	flips (w/b)
Llama-3.2-1B	Q4_K_M	MMLU	500	0.442	0.430	+1.2	34/28
Llama-3.2-1B	Q8_0	MMLU	500	0.442	0.436	+0.6	7/4
Llama-3.2-3B†	Q4_K_M	MMLU	500	0.628	0.638	-1.0	19/24
Llama-3.2-3B†	Q8_0	MMLU	500	0.628	0.626	+0.2	2/1
Qwen3-1.7B	Q4_K_M	MMLU	500	0.548	0.520	+2.8	36/22
Qwen3-1.7B	Q8_0	MMLU	500	0.548	0.536	+1.2	13/7
Qwen3-8B†	Q4_K_M	MMLU	500	0.686	0.596	+9.0	73/28
Qwen3-8B	Q8_0	MMLU	500	0.686	0.680	+0.6	6/3
Llama-3.2-1B	Q4_K_M	GSM8K	300	0.440	0.413	+2.7	35/27
Llama-3.2-1B	Q8_0	GSM8K	300	0.440	0.400	+4.0	21/9
Qwen3-1.7B	Q4_K_M	GSM8K	300	0.033	0.020	+1.3	10/6
Qwen3-1.7B	Q8_0	GSM8K	300	0.033	0.033	+0.0	9/9

7 Results

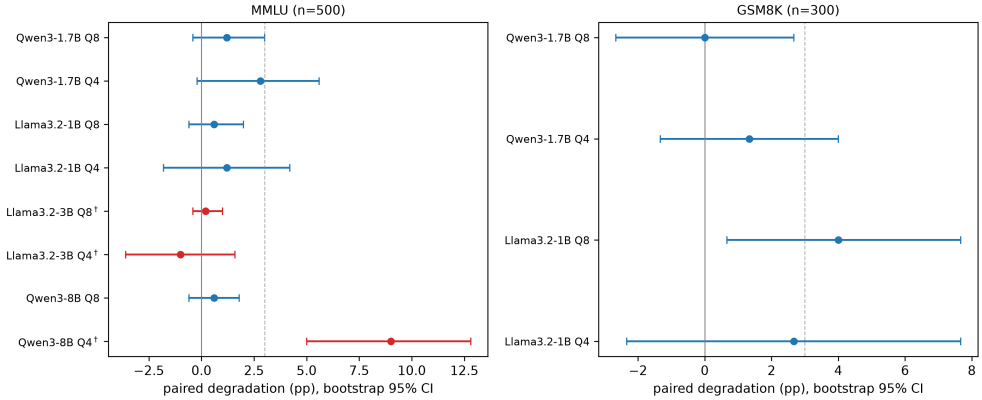


Fig. 1 Measured paired degradation across deployment configurations. Error bars are paired bootstrap 95% confidence intervals (4000 resamples). † marks a cross-machine/build reference–candidate pair (see §7.8). Intervals are descriptive item-resampling summaries, not R2 certificate bounds.

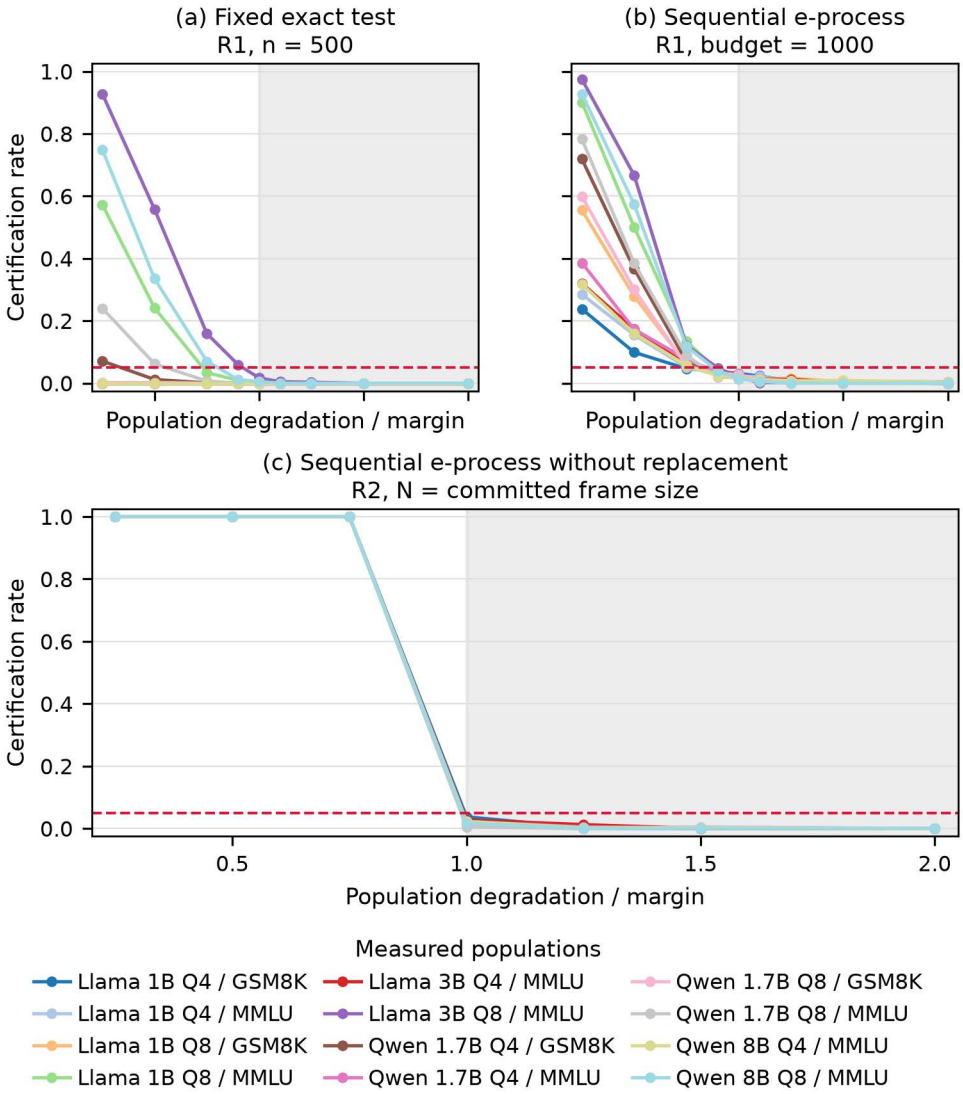


Fig. 2 Certification rate versus population degradation (as a fraction of the margin) for all twelve measured populations, with population degradation swept through the boundary by controlled relabeling of real flip records; the shaded area is the null region $\Delta \geq \varepsilon$ (ε=3%, δ=5%). (a) Exact binomial fixed test, R1, n=500, 1000 replicates/cell. (b) R1 sequential i.i.d. e-process, optional stopping, budget 1000. (c) R2 without-replacement e-process with terminal census rule on the finite frame (N=500 MMLU, N=300 GSM8K), 400 permutations/cell — note the sharper certify-to-abstain transition and full power below the margin from frame exhaustion. All curves stay below δ throughout the null region.

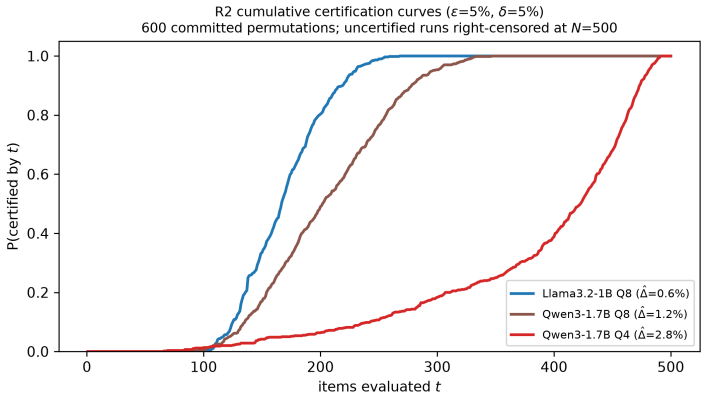


Fig. 3 R2 cumulative certification curves (empirical CDF of the stopping time) at ε=5%, δ=5%: proportion of 600 committed random permutations certified by t evaluated items (without replacement, N=500). Medians: 167 (Llama-3.2-1B Q8, $\hat{\Delta}$=0.6%), 202 (Qwen3-1.7B Q8, $\hat{\Delta}$=1.2%), 423 (Qwen3-1.7B Q4, $\hat{\Delta}$=2.8% — the higher-discordance, harder configuration). Runs not certified by N are right-censored at N.

7.1 Measured degradation is task- and configuration-dependent

Table 3 and Figure 1 summarize the paired degradation across configurations. Three structural facts matter for certification. First, degradation varies by an order of magnitude across configurations that practitioners treat as interchangeable. Among same-machine/build rows, Q8_0 on MMLU costs 0.6–1.2 percentage points and Q4_K_M 1.2–2.8pp — the latter near a 3pp illustrative tolerance (ratio 0.93). Cross-machine/build rows span −1.0pp (an apparent small improvement, Llama-3.2-3B Q4) through +0.2pp (Llama-3.2-3B Q8) to +9.0pp (Qwen3-8B Q4), illustrating configuration confounding rather than quantization loss alone. Second, flips are bidirectional and can dwarf the net: on GSM8K, Llama-3.2-1B Q4 flips 11.7% of items worse and 9.0% better for a net of only 2.7pp, and Qwen3-1.7B Q8 flips 3% each way for a net of exactly zero — point-estimate comparisons hide item-level discordance up to 21%. Third, discordance rates (0.6–21%) directly govern the paired design's variance advantage: across the eight MMLU configurations the paired-difference variance is 2.4–78× smaller than the corresponding unpaired two-sample variance at equal inference budget, largest at the lowest discordance, as quantified in the problem formalization (§3).

7.2 Error-rate preservation on measured populations

Figure 2 shows certification rate versus empirical-population degradation (as a fraction of tolerance) for all twelve measured populations, with population degradation swept through the boundary by controlled relabeling of real flip records. This is an *implementation validation on measured flip distributions*: the operating characteristics are Monte Carlo *estimates* on the empirical populations treated as ground truth (the formal guarantees are Theorems 1–3 and Corollary 4), and relabeled boundary populations need not preserve real error mechanisms. Across all null-region (inferior-at-or-beyond-margin) population cells ($\Delta \geq \varepsilon$; grid over $\varepsilon \in \{2, 3, 5\}\%$, $\delta = 5\%$), none of the proposed procedures exceeded its budget in any cell (Table 4).

Table 4 Observed null-cell error rates for the proposed procedures.

Regime	Instrument	Stopping policy	Worst observed null rate
R1	exact binomial fixed	$n = 500$	2.9%
R1	e-process (terminal)	fixed $n = 500$, $W_n \geq 1/\delta$ at n	0.5%
R1	e-process	optional stopping (budget 1000)	3.3%
R2	WoR e-process + census	optional stopping (N = frame)	3.8%

(All four rows are computed by the shipped drivers on the current populations: the three R1 arms by `mc_validation.py --mode r1-full` *— whose null-budget assertion covers every arm — and the R2 row by* `--mode full`*; the fixed-n e-process row is the* terminal *test* $W_{500} \geq 1/\delta$ *evaluated at* n *exactly, with no interim-crossing credit.)*

Controlled relabeling procedure.

Each grid cell's population is constructed from a measured flip record $d \in \{-1, 0, +1\}^N$ as follows. Let $k = \text{round}\big((\Delta_{\text{target}} - \bar{d})\, N\big)$. If $k > 0$, relabel k uniformly chosen concordant items ($d_i = 0$) to $+1$; if $k < 0$, relabel $|k|$ uniformly chosen harmful flips ($d_i = +1$) to 0; beneficial flips ($d_i = -1$) are never modified. The construction therefore preserves the measured beneficial-flip count exactly, changes total discordance $p_w + p_b$ by exactly $|k|/N$ (monotonically in the shift direction), and realizes the target mean up to rounding to the $1/N$ grid (the realized mean, not the nominal target, defines the cell's regime). Relabeling uses a fixed master seed per driver, and R1 and R2 cells at the same (population, Δ/ε) coordinate share the same *outcome distribution*: the relabeled worse/better/concordant counts are identical, which fully determines both the i.i.d. sampling distribution (R1) and the permutation distribution (R2). The two drivers consume their RNG streams differently, so itemwise relabeled arrays are not forced to coincide — the shared object is the count vector, not the array.

(R1 grid: 324 cells — 180 null, 144 alternative — 1000 replicates each; R2 grid: 252 cells — 144 null, 108 alternative — 400 permutations each, regenerated end-to-end by the shipped driver.) The regression checks require null-cell estimates to remain within a prespecified three-Monte-Carlo-standard-error tolerance of the nominal budget. The R2 checks also require every alternative cell at census-guaranteed rate exactly 1. At the exact boundary $\Delta = \varepsilon$ all procedures are conservative (exact test 0.0–2.9%; R1 sequential 1.0–3.3%; R2 0.5–3.8%; worst R2 null cell overall 3.8%). These observed rates are point estimates: a cell observed at 3.8% over 400 replicates has a one-sided 95% Clopper–Pearson upper bound of 5.7%, so the simulations are consistent with — but do not by themselves prove — the δ level; they are implementation evidence agreeing with the theorems, not an independent demonstration of nominal control. Below the margin, R2 power is exactly 1 in every alternative cell — a *structural consequence of the terminal census rule* (the frame mean is known at $t{=}N$), not an empirical achievement of the e-process, whose contribution below the margin is early stopping, not the eventual decision. The certify/abstain transition at the boundary is a step, not a slope (Figure 2, right panel). The conservatism of the exact test is the price of its unconditional validity: on higher-discordance populations (GSM8K, discordance up to 21%) it certifies clearly non-inferior configurations at $n{=}500$ with power near zero, while the e-process retains usable power on the same populations — the operational reason we recommend the e-process as the primary certification instrument.

7.3 Early stopping under the without-replacement protocol

Figure 3 shows R2 cumulative certification curves (the empirical CDF of the certification time) at $\varepsilon{=}5\%$ over 600 committed random permutations of the 500-item frame.

In random-order simulations and retrospective replays, clearly non-inferior configurations often crossed before the committed frame cap. The process crosses in a median of 167 items (Llama-3.2-1B Q8, $\hat{\Delta}$=0.6%) and 202 items (Qwen3-1.7B Q8, $\hat{\Delta}$=1.2%); the higher-discordance configuration (Qwen3-1.7B Q4, $\hat{\Delta}$=2.8%, discordance 11.6%) certifies in a median of 423 items. Because the finite frame is exhausted at N, the R2 process eventually certifies every truly non-inferior population — unlike its i.i.d. counterpart — and abstention within the frame is concentrated on populations near or beyond the margin. The stopping distributions are reported as numbers of evaluated items, not as a prospective cost-saving trial or a power-matched economic comparison. Against standard without-replacement baselines the picture is consistent (400 permutations per cell, seed 20260906, ε=3%, four populations spanning low- and high-discordance): all three anytime-valid WoR procedures hold the null within Monte Carlo resolution (boundary rates: betting mixture 0.8–2.3%, Hoeffding-type 0.0%, empirical-Bernstein-style 3.8–6.0%, the upper endpoint within one MC standard error of δ at 400 replicates) and certify every clearly non-inferior population; on low-discordance MMLU frames the variance-adaptive empirical-Bernstein bets stop earliest (median 268.5–285 items), the betting mixture is close behind (317–339.5), and the Hoeffding-type bet — tuned to worst-case variance — rarely stops before the census (491). On the GSM8K frames (N=300) the ordering persists but compresses: at the clearest ratio the medians are 253/295/230 (mixture/Hoeffding/EB-style) on one frame — the gaps narrow as the census cap approaches, but the methods do not coincide. The mixture's advantage over a single tuned bet is robustness: it needs no variance pilot, and across the four measured cells in this comparison it was close to the best-performing arm. (Both baselines bet on the same conditional-mean payoff $X_t = m_t - d_t$ with the census rule; all tuning constants are fixed a priori, before any outcome is observed. The Hoeffding-type procedure uses the fixed bet $\lambda = \sqrt{8\log(1/\delta)/(4n^*)}$, $n^* = N/2$, with the Hoeffding supermartingale correction $\exp(\lambda X_t - \lambda^2/2)$ for range-2 payoffs. The variance-adaptive procedure is an *empirical-Bernstein-style variance-adaptive betting baseline* (not the classical closed-form EB confidence sequence), defined exactly as: running moments initialize at $\hat{\mu}_0 = 0$, $\hat{v}_0 = 1$ (a conservative range-scale prior); at step t the predictable bet is $\lambda_t = \min\{\sqrt{2\log(1/\delta)/(\max(\hat{v}_{t-1}, 10^{-4}) \cdot \max(t, 2))},\ c/(1-m_t)\}$ with $c = 1/2$, computed *before* observing d_t (clipping applied after the variance term); log-wealth accumulates $\log(1 + \lambda_t X_t)$ and certifies on crossing $\log(1/\delta)$; after the update, the running mean and variance of the observed d's advance by Welford's recurrences with denominator t. The same pseudocode ships as executable source in the artifact.)

To separate the stopping policy from the statistic, we also evaluated the *same* mixture e-process at a single fixed horizon ($n = N$, no optional stopping) and a *simulation-calibrated net-count baseline* (statistic $n_w - n_b$; note it collapses the two counts to their difference and discards total discordance $n_w + n_b$): reject when $n_w - n_b$ falls below the δ-quantile of its null distribution, calibrated by Monte Carlo over a 9-point grid of (p_w, p_b) with $p_w - p_b = \varepsilon$, taking the worst case over the grid. Because the composite-null supremum is approximated on a finite grid with Monte Carlo error, we use this baseline only as an *exploratory power comparison*, not as a level-guaranteed test; a rigorous unconditional version would optimize the supremum p-value over the full null region with certified numerical error. On a clearly non-inferior low-discordance

R1 population ($\Delta = 1\%$, $\varepsilon = 3\%$, $n = 500$), 2000 replicates gave power 0.31 for the sequential e-process, 0.18 for the fixed-horizon e-process, 0.00 for the exploratory net-count baseline, and 0.16 for the harmful-flip-only exact test — all conservative at this margin-gap-to-discordance ratio, while the corresponding R2 frame certifies with probability one by census. Two conclusions follow: the sequential advantage over the identical fixed-horizon statistic at equal n is modest — the value of optional stopping is stopping *early* on clear cases, not extra terminal power — and the harmful-flip-only test loses power by ignoring beneficial flips, with discreteness contributing additional conservatism; the exploratory net-count baseline is itself highly conservative under worst-case null calibration and should not be read as an optimized paired test. A power-matched comparison against group-sequential designs remains future work. Runs not certified by N are right-censored in Figure 3, and a boundary "median N" means at least half the runs did not certify within the frame.

7.4 Within-configuration determinism

The paired-record design assumes each configuration maps an item to a deterministic outcome. We verified this on four Jetson configurations (greedy decoding, batch 1, fixed seed, same build): Qwen3-1.7B Q4_K_M on CUDA (4 runs), BF16 on CUDA (3 runs), Q8_0 on CUDA — the example certificate's candidate configuration — (3 runs), and Q4_K_M on the CPU backend (4 runs). Every run pair reproduced *identical* parsed answers on all 500 items — identical accuracies, identical flip sets. Parsed-answer identity is computed on the parsed answer token after whitespace normalization, with the parser hash a signed certificate field (§5); it is not a raw-generation-string comparison. Two Mac-side (x86-64 CPU build) configurations were also repeated: Qwen3-8B Q4_K_M (2 runs) and Qwen3-1.7B Q4_K_M — the model of the cross-machine comparison in §7.8 — (3 runs); both reproduced identical parsed answers on all 500 items. One provenance limitation is relevant: the Mac repeats ran on a rebuilt llama.cpp (via llama-cpp-python 0.3.35) because the original Mac build was not preserved, so they verify determinism of the *current* Mac configuration, not byte-level identity with the original run. Consistent with the paper's thesis, the rebuild is itself a different configuration: the same 1.7B Q4 artifact under the rebuilt engine shifts accuracy by 1.2pp against the original Mac run — an engine/build configuration change moved item-level behavior (the original build was not preserved, so the change cannot be attributed to the engine version alone); this rebuilt configuration is the primary cross-machine comparison of §7.8. Across all six repeated configurations — spanning two machines, two backends (CUDA and CPU), two model sizes, and three precisions — no execution noise was observed. A GSM8K repeat (Qwen3-1.7B Q4, CUDA, two runs over all 300 items, executed under the v2 harness) produced identical parsed predictions — parser-level repeatability. Because the v2 harness stored neither the full generation nor the stop reason, this repeat does not by itself establish full-generation equality or v3 (truncation-aware) repeatability: distinct generations can parse to the same prediction. The v3 rescoring is a deterministic function of its own stored generations, covering the long-generation regime.

7.5 Resource–quality trade-off on the edge device

The certificate answers "is the quality loss within tolerance"; the release decision also needs the benefit side. Table 5 reports the measured trade-off for the Qwen3-1.7B family on the Jetson Orin Nano: observed end-to-end generated-token throughput medians (generated tokens divided by total request wall time, which includes prompt processing — not the server's decode-only telemetry) from the GSM8K generation runs (items generating >50 tokens), MMLU per-item latency medians over all 500 items, and GGUF file size as an artifact-size measure; it is not a measurement of peak resident memory, which additionally includes the KV cache, allocator state, and backend-specific workspaces (peak RSS was not measured, so reductions are model-file-size reductions). Their token-throughput measurements characterize the measured generation workload rather than task-effective throughput.

Config (CUDA)	GGUF	tok/s	item lat.	$\hat{\Delta}$	Retrospective replay: T and $\widetilde{U}_T$
BF16 (ref)	3.4 GB	18.5	0.51 s	—	—
Q8_0	1.8 GB	30.0	0.34 s	+1.2pp	replay crossing‡ T=283, $\widetilde{U}_T$=0.049851
Q4_K_M	1.1 GB	34.0	0.31 s	+2.8pp	replay crossing‡ T=426, $\widetilde{U}_T$=0.049973

Table 5 Resource–quality trade-off, Qwen3-1.7B on Jetson Orin Nano (llama.cpp CUDA, greedy, batch 1). Both quantized configurations are evaluated at the committed margin $\varepsilon = 5\%$ under R2 on the 500-item frame. In retrospective protocol replay (‡: not a prospectively anchored release event), Q8_0 offers a 1.9× model-file-size reduction and 1.6× generated-token–throughput speedup, crossing at $T = 283$; Q4_K_M buys a further 1.6× file-size reduction (3.1× cumulative vs. BF16) and crosses at $T = 426$ at the same margin. At the tighter $\varepsilon = 3\%$ (frame mean 2.8pp, ratio 0.93) Q4_K_M certifies only at the end of the frame ($T = 499$ on the committed permutation) — no early-stopping benefit near the boundary. Electrical power draw and energy were not measured; the device was operated in its fixed 15 W power mode. Latency medians are over 500 (MMLU) items; generated-token throughput is the median over the *same* 300 GSM8K items for all three configurations (every item generated >50 tokens in every configuration, so the intersection filter excludes nothing and workload composition is identical across rows).

7.6 The boundary case exists in practice

Qwen3-1.7B Q4_K_M on MMLU ($\hat{\Delta} = 2.8\%$) is precisely the configuration class that motivates certification: at ε=3% its degradation-to-tolerance ratio is 0.93, and the two regimes behave differently there in exactly the way the theory predicts. Under R1 (a fresh audit sampled from this population), uniformly valid procedures necessarily face low power this close to the margin, and the sequential test usually abstains within any practical budget. Under R2, certification is guaranteed no later than the census at $N = 500$ (the frame mean is $2.8\% < 3\%$, Corollary 4); in the same retrospective replay the deterministic null-impossibility check (the $m_t > 1$ branch of Theorem 3) certified one item earlier, after $T = 499$ observed items — the mixture wealth itself never approaches the threshold at this margin. What the near-boundary case costs, in R2, is essentially the entire frame rather than the release. Under R1, point-estimate

practice remains a coin flip: treating the measured flip distribution as the population, a single 500-item bootstrap evaluation shows $\hat{\Delta} \leq 2\%$ with probability 32% (multinomial over $(p_w, p_b) = (0.072, 0.044)$) and would ship or reject on sampling luck. Under R2, evaluating the entire committed frame makes its mean exact, but a bare point estimate provides neither valid early stopping nor a predeclared release-risk guarantee. The protocol replaces these two failure modes with calibrated abstention under R1 and an anytime-valid process with an exact census fallback under R2.

7.7 Candidate selection inflates risk; risk-splitting restores control

Release pipelines rarely evaluate one candidate: teams compare several quantization levels and pick the best-looking one. We simulated this pipeline on measured populations under a specified candidate-dependence model: $K \in \{1, 2, 4, 8\}$ candidates, each an independent draw of n=500 items (with replacement, R1) from a distinct measured MMLU population (fixed candidate-to-population mapping recorded in the driver manifest) relabeled to sit exactly at the boundary ($\Delta = \varepsilon = 3\%$, none certifiable); the team selects the candidate with the lowest observed $\hat{\Delta}$ before certifying (2000 replicates; sequential e-process over the 500-item sample — certification horizon 500, no additional draws; the holdout policy selects on the first 250 items of each candidate's sample and certifies on a fresh independent 500-item draw). Under R2, a precommitted holdout certifies the mean of the *held-out audit subframe*; it does not certify the combined selection-plus-holdout frame without an additional selection-aware argument. For $K \geq 2$, disjointness alone is insufficient: the selected identity must be fixed from selection evidence before the audit permutation is revealed, and fresh Stage-2 randomness must then be conditionally independent as stated in Assumption 1. Choosing among null candidates after seeing the common audit order can make the original R2 argument fail. Protocol 3.1 therefore enforces the chronology rather than claiming a new general holdout theorem. Because candidate samples are drawn independently, the naive-inflation curve is illustrative — real quantization candidates share items and their errors correlate, which changes the curve's shape but not the validity of the δ/K or holdout mitigations, whose union-bound logic is correlation-free. The exact procedure, fixed for reproducibility:

```
for rep in 1..2000:
  for k in 1..K: draw S_k = 500 items iid from population k
  k^full = argmin_k Δ̂(S_k[1:500])
  naive: run e-process on S_{k^full} (same 500 records replayed; horizon 500)
  δ/K: same records for k^full, threshold K/δ
  k^half = argmin_k Δ̂(S_k[1:250])   # fresh selection on half
  holdout: certify k^half on a fresh independent 500-item draw (horizon 500)
report fraction certified per arm
```

In a finite-frame (R2) deployment "fresh draw" means a disjoint committed audit frame whose random inspection order is finalized only after selection; the observed remainder of a frame already used for selection is not fresh, and a precommitted split does not cure audit-order leakage if that later order was already revealed. Certifying the selected candidate *on the same sample* inflates the false-release rate from 1.2% (K=1) to 7.4% (K=8) — past the $\delta = 5\%$ budget — because selection favors

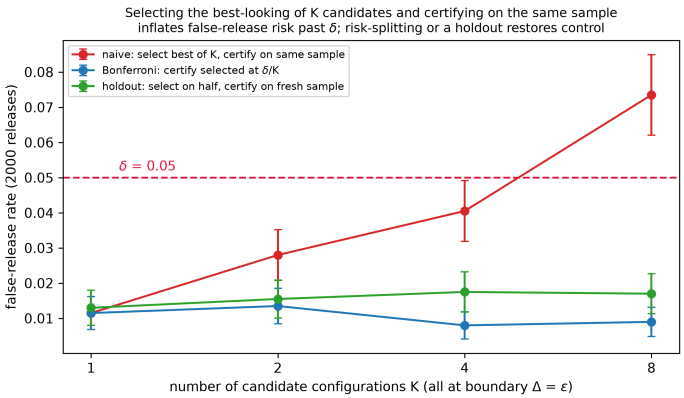


Fig. 4 False-release rate when the best-looking of K boundary candidates is selected and then certified, under three certification policies (2000 simulated releases per point; error bars are 95% binomial CIs).

candidates with lucky draws. Both mitigations restore control at all K: certifying the selected candidate at level δ/K (observed false-release rates at or below 1.4% for $K \geq 2$), or selecting on one half of the data and certifying on a fresh sample (observed rates at or below 1.8%) (Figure 4). The protocol therefore mandates one of these whenever certification follows a comparison. Because releases repeat — new builds, new model versions, retried comparisons — per-release δ-budgets also compound over a release family's lifetime. The commitment's risk allocation therefore implements a lifetime ledger $\sum_r \delta_r \leq \delta_{\text{lifetime}}$ (our reference implementation defaults to $\delta_r = \delta_{\text{lifetime}}/2^r$). Within one signed release, K is the count of complete profiles in the committed universe and is the denominator used by the Bonferroni check. This binds the supplied universe but cannot prove that the issuer disclosed every candidate it screened.

Proposition 7 (Lifetime ledger) *Suppose release r is conducted at level δ_r, where δ_r is predictable (fixed before release r's outcomes are observed) and $\sum_r \delta_r \leq \delta_{\text{lifetime}}$, with any within-release candidate selection budgeted so that release r's false-certification probability given its history is at most δ_r (e.g. $\sum_{k \in \mathcal{K}_r} \delta_{r,k} \leq \delta_r$). Then by a union bound over releases, the probability of at least one false certification over the family's lifetime is at most δ_{lifetime}.*

Lifetime control therefore holds only within an externally governed release family. Protocol 3.1 signs the family, release index, declared parent id/digest, this release's risk, lifetime budget, and the geometric allocation rule, and it checks $\delta_r \leq \delta_{\text{lifetime}}/2^r$. Its current verifier does not ingest or certify a complete family registry, so it cannot rule out siblings, omitted ancestors, undeclared families, or a fresh signing identity; the certificate says that its history scope is current-commitment-only. An append-only registry can supply the missing governance premise. Retained historical compatibility objects do not establish that premise. The default allocation $\delta_r = \delta_{\text{lifetime}}/2^r$ runs over $r = 1, 2, \ldots$ and sums to δ_{lifetime} if the externally governed family follows it.

7.8 Certification must bind to configurations, not model files

Figure 5 presents the cross-configuration measurements, and a 2×2 factorial on a single device resolves the attribution question they raise. *Factorial design*: Qwen3-1.7B, the same 500 MMLU items, the same Jetson device and llama.cpp build, crossing precision (BF16 vs. Q4_K_M) with execution backend (CUDA, `ngl`=99 vs. CPU, `ngl`=0). The quantization main effect is +2.9pp (paired bootstrap 95% CI $[-0.1, +5.8]$pp; simple effects $+2.8 \pm 3.0$pp on CUDA, $+3.0 \pm 3.0$pp on CPU); the backend main effect is −0.1pp (CI $[-0.5, +0.2]$pp) and the interaction −0.2pp (CI $[-1.0, +0.4]$pp) — both CIs tightly bracket zero. Within a fixed machine and build, greedy predictions agree across backends on 99.6% (BF16; Wilson 95% CI $[98.6, 99.9]$) and 98.8% (Q4; CI $[97.4, 99.4]$) of items, versus 88.2% (CI $[85.1, 90.7]$) for the same Q4 artifact across machines. In this model/machine pair, the machine/build change was materially larger than any within-machine backend change; broader hardware-class claims would require sampling multiple physical units, so certificates remain device- and configuration-specific. These intervals are descriptive item-resampling summaries and are not the R2 certificate bound or confidence sequence for the committed finite audit frame.

Contrast this with the *cross-machine* comparison. This is an archived outcome-record comparison using a rebuilt Mac configuration; the original historical Mac build was not preserved. It evaluates the identical Q4_K_M GGUF artifact on the Jetson (aarch64, CUDA build) versus the Mac (x86-64, CPU build of llama.cpp via llama-cpp-python 0.3.35, deterministic over 3 repeats): the two configurations agree on only 88.2% of predictions, with per-machine accuracies 52.0% vs. 53.2% and flip sets overlapping at Jaccard 0.47 (22 items flip only on the Jetson, 18 only on the Mac, 36 on both). An earlier, incompletely archived Mac build showed the same qualitative phenomenon (90.6% agreement, Jaccard 0.54); that the two Mac builds themselves disagree (1.2pp accuracy shift on the same artifact and items) is itself an instance of the paper's thesis — an engine/build configuration change moved item-level behavior. All headline cross-machine numbers use the rebuilt comparison records. Together the two experiments localize the source of configuration-specificity: the observed precision-by-backend interaction within this machine was small relative to the cross-machine disagreement (the factorial was not designed as an equivalence test), while changing the machine and engine build shifts behavior materially even at fixed precision. Certificates must therefore bind to the hardware target and engine build — the fingerprint fields of §5 — and a certificate issued on one machine class is silent about another. Similarly, the 8B contrast in Table 3 (Q8 same-machine +0.6pp vs. Q4 cross-machine +9.0pp) confounds quantization level with the machine change; the 1.7B factorial is consistent with the compound configuration change being important, but does not identify the decomposition of the 8B gap; a same-machine 8B factorial (infeasible at BF16 on an 8 GB device) would be required to close the attribution fully.

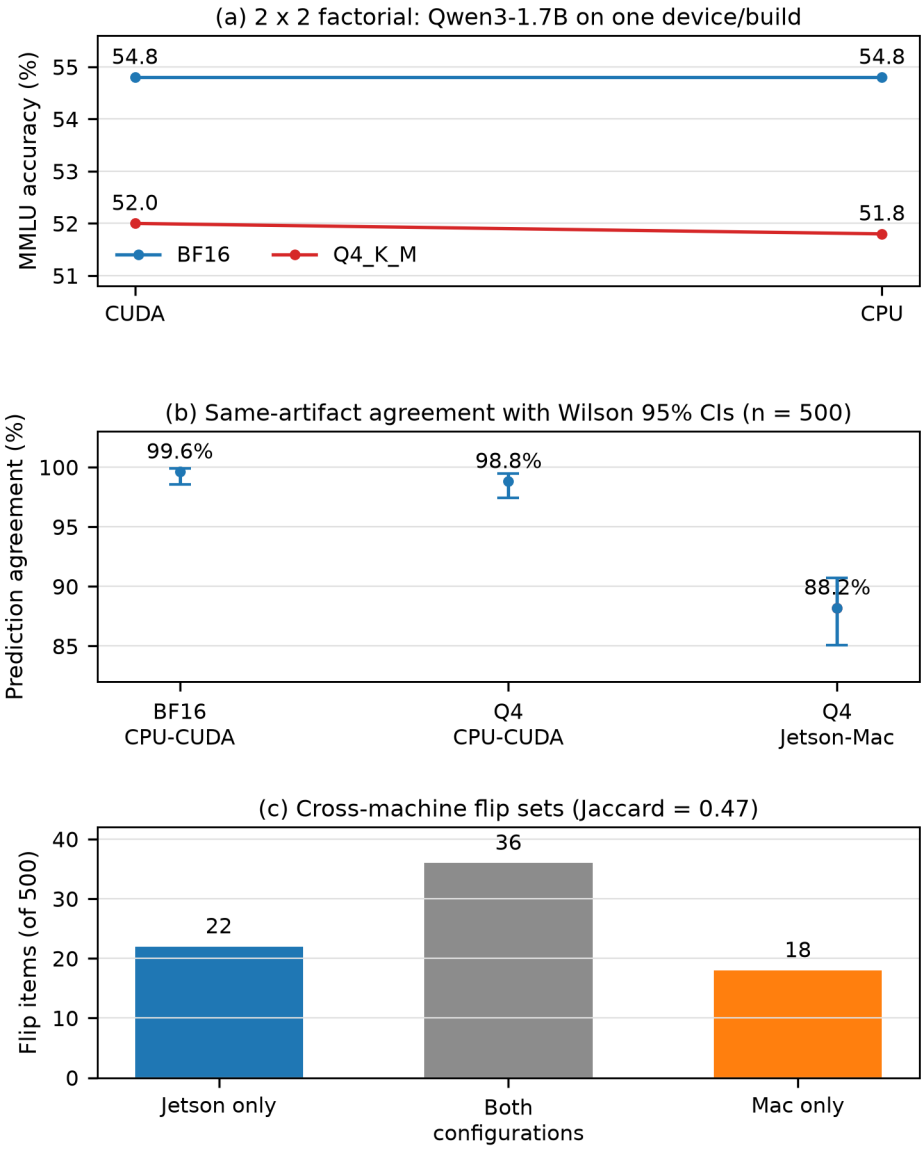


Fig. 5 (a) 2×2 factorial on one device and build (Qwen3-1.7B, 500 MMLU items): quantization main effect +2.9pp, backend main effect −0.1pp, interaction −0.2pp (paired bootstrap CIs in §7.8; resampling is over item identifiers, jointly resampling each item's four configuration outcomes to preserve within-item dependence). (b) Same-artifact greedy-prediction agreement with Wilson 95% CIs: near-total within a machine, 88.2% across machines (rebuilt Mac configuration; archived outcome records). (c) Cross-machine flip sets relative to the shared reference overlap at Jaccard 0.47.

8 Discussion and Limitations

What certification does not claim.

This certificate establishes relative non-inferiority to an independently accepted reference configuration; it is not an absolute task-suitability or production-readiness certificate. The numerical margins used in this study illustrate the protocol; production margins must be justified by application-specific quality requirements. Under R1, a certificate concerns the declared task distribution; under R2, it concerns only the committed finite audit frame. In both regimes, the statement is specific to the recorded deployment configuration. It does not bound behavior off-distribution, under adversarial inputs, outside the committed R2 frame, or after any component of the serving path changes. We view this narrowness as a feature: the certificate says exactly what was tested and how confidently. Protocol 3.1 verifies the integrity and contract completeness of the supplied deployment profiles and the digital correspondence of actual supplied task, evaluator, frame, selection, and record files. It does not verify physical execution, remote attestation, truth of self-signed timestamps or claimed beacon values, trustworthiness of the supplied public key, completeness of unseen candidates or release history, or absence of precommit screening. Those require external attestation, anchoring, and governance.

Archive and provenance boundary.

The public artifact supports certificate-decision replay and paired-correctness reconstruction: complete retained reference–candidate correctness pairs exist for eight of the twelve populations, and four populations support derived-array downstream replay because the relevant historical reference runs were not preserved. The public artifact supports independent reconstruction of paired degradation from retained correctness records, but not independent re-parsing or re-scoring of the original generations. The package does not reconstruct every historical execution environment; in particular, the Mac comparison is an archived outcome-record comparison using a rebuilt Mac configuration, and the original historical Mac build was not preserved. Replay is not physical-execution attestation. A candidate manifest freezes the supplied candidate universe but cannot prove that no hidden pre-screening occurred.

Truncation and metric design.

On generation tasks with bounded output budgets, truncation is part of the deployed behavior. The metric scores a budget-exhausted generation (`stop_type = limit`) as failure regardless of any parseable partial answer; alternative conventions (extend budget, judge partial answers) change m, not the protocol. All reported GSM8K accuracy, correctness-status change, and grid results use this truncation-as-failure rule. The determinism repeats reported in Section 7.4 predate this metric implementation and establish parsed-prediction repeatability only, not truncation-aware repeatability.

Judge-based metrics.

Judge-based metrics introduce additional measurement noise and potential systematic bias that are not covered by the present validation.

Scope of validation.

Our populations come from two model families, two tasks, two quantization levels, and Jetson and Mac deployment configurations. The guarantees are nonparametric: R1 is valid for bounded i.i.d. outcomes, whereas R2 applies to bounded fixed finite populations under uniform random inspection, subject to the stated pre-specification assumptions. The empirical question is whether measured degradation distributions are representative in structure (discordance rates, flip asymmetry); our measured configurations span a broad range, from 0.6% to 21% discordance — low-, moderate-, and higher-discordance regimes.

Relation to runtime observability.

Configuration-level certificates and runtime risk gating (Wei et al. 2026; Wei and Liu 2026) compose: the certificate gates the release; runtime monitors detect drift and trigger re-certification. Drift monitoring is a separate statistical problem from release certification — once the input distribution shifts, the certified estimand itself changes — so the lifecycle couples a release certifier (this paper) with a separately-budgeted drift detector whose alarm supersedes the standing certificate and triggers a fresh audit sample.

9 Conclusion

This work supplies a bounded software-quality release gate for deciding whether a recorded quantized-LLM deployment configuration may replace a recorded reference under a declared non-inferiority margin and false-certification budget. It combines paired fixed-sample and anytime-valid procedures with configuration and evidence binding, explicit candidate-selection and lifetime-risk rules, certificate-decision replay, and a re-certification lifecycle. The measured deployment study tests the implementation and illustrates why evidence must be configuration-specific. The paper proposes neither a new quantization method nor new general-purpose statistical theory; its contribution is the end-to-end assurance workflow and the precise scope of the resulting R1 distribution or R2 finite-frame decision. This is relative release evidence, not absolute task suitability, production readiness, or full product approval.

Supplementary materials

Online Resource 1 provides the default commands, schema and fixture mappings, protocol versions and compatibility, deterministic-driver specifications, environment information, data provenance, and the negative-fixture catalogue. Online Resource 2 provides the protocol schemas, issuer and replay verifier, synthetic positive and negative fixtures, analysis and responsibility drivers, retained outcome records, and authoritative derived arrays. It excludes model weights, original-generation text, and unnecessary copies of benchmark question or answer text.

Statements and Declarations

Acknowledgements.

None.

Author contributions.

M.J. and H.L. co-led the analysis, interpretation of the results, and manuscript preparation. H.L. conceived the study and led the methodology, software/certification framework, and computational implementation. M.J. contributed to study design, validation, integration, and supervision. D.K., C.S., M.P., and I.-H.Y. contributed to implementation, validation, data analysis, and manuscript review. All authors reviewed and approved the final manuscript.

Funding.

M.J. acknowledges support from the Institute of Information & Communications Technology Planning & Evaluation (IITP) grant IITP-2026-RS-2024-00437191 and the National Research Foundation of Korea (NRF) grants RS-2018-NR031074 and RS-2023-00256050, funded by the Korean government.

Competing interests.

The authors have no relevant financial or non-financial interests to disclose.

Data availability.

The supplementary artifact contains the retained evidence required to reproduce the reported certificate decisions and downstream paired-degradation analyses. Complete retained reference-candidate correctness records are available for eight of the twelve measured populations; for the remaining four populations, authoritative derived arrays and summaries are provided. The public artifact supports independent reconstruction of paired degradation from retained correctness records, but not independent re-parsing or re-scoring of the original generations.

Code availability.

The source code supporting the certification and verification workflow is provided in the supplementary artifact accompanying this article. The released source code is distributed under the MIT License.

Originality and dual publication.

The results, data, and figures reported in this manuscript have not been published elsewhere and are not under consideration by another publisher.

Third-party material.

No third-party material requiring separate permission.

Generative-AI disclosure.

During manuscript preparation and internal verification, generative-AI tools were used to assist code review, consistency checking, and language revision. All suggested text and code changes were reviewed and validated by the authors, who take full responsibility for the manuscript, analyses, and released artifact.

References

Adams, C., Cain, P., Pinkas, D., Zuccherato, R.: Internet X.509 Public Key Infrastructure Time-Stamp Protocol (TSP). RFC 3161 (2001). https://doi.org/10.17487/RFC3161 . https://www.rfc-editor.org/info/rfc3161

Ashmore, R., Calinescu, R., Paterson, C.: Assuring the machine learning lifecycle: Desiderata, methods, and challenges. ACM Computing Surveys **54**(5), 1–39 (2021) https://doi.org/10.1145/3453444 . Article 111

Breck, E., Cai, S., Nielsen, E., Salib, M., Sculley, D.: The ML test score: A rubric for ML production readiness and technical debt reduction. In: 2017 IEEE International Conference on Big Data, pp. 1123–1132 (2017). https://doi.org/10.1109/BigData.2017.8258038

Cobbe, K., Kosaraju, V., Bavarian, M., Chen, M., Jun, H., Kaiser, L., Plappert, M., Tworek, J., Hilton, J., Nakano, R., Hesse, C., Schulman, J.: Training verifiers to solve math word problems. arXiv:2110.14168 (2021). https://doi.org/10.48550/arXiv.2110.14168 . https://arxiv.org/abs/2110.14168

Frantar, E., Ashkboos, S., Hoefler, T., Alistarh, D.: GPTQ: Accurate post-training quantization for generative pre-trained transformers. In: The Eleventh International Conference on Learning Representations (2023). https://openreview.net/forum?id=tcbBPnfwxS

Gerganov, G., et al.: llama.cpp: LLM inference in C/C++ and the GGUF format. GitHub repository (2023–). https://github.com/ggml-org/llama.cpp

Grattafiori, A., et al.: The Llama 3 herd of models. arXiv:2407.21783 (2024). https://doi.org/10.48550/arXiv.2407.21783 . https://arxiv.org/abs/2407.21783

Hendrycks, D., Burns, C., Basart, S., Zou, A., Mazeika, M., Song, D., Steinhardt, J.: Measuring massive multitask language understanding. In: The Ninth International Conference on Learning Representations (2021). https://openreview.net/forum?id=d7KBjmI3GmQ

Josefsson, S., Liusvaara, I.: Edwards-Curve Digital Signature Algorithm (EdDSA). RFC 8032 (2017). https://doi.org/10.17487/RFC8032 . https://www.rfc-editor.org/info/rfc8032

Kübler, J., Budhathoki, K., Kleindessner, M., Zhou, X., Yin, J., Khetan, A., Karypis,

G.: When LLMs get significantly worse: A statistical approach to detect model degradations. In: The Fourteenth International Conference on Learning Representations (2026). ICLR 2026 conference paper; arXiv:2602.10144. https://openreview.net/forum?id=cM3gsqEI4K

Lin, J., Tang, J., Tang, H., Yang, S., Chen, W.-M., Wang, W.-C., Xiao, G., Dang, X., Gan, C., Han, S.: AWQ: Activation-aware weight quantization for on-device LLM compression and acceleration. In: Proceedings of Machine Learning and Systems, vol. 6, pp. 87–100 (2024). https://proceedings.mlsys.org/paper_files/paper/2024/hash/42a452cbafa9dd64e9ba4aa95cc1ef21-Abstract-Conference.html

Rundgren, A., Jordan, B., Erdtman, S.: JSON Canonicalization Scheme (JCS). RFC 8785 (2020). https://doi.org/10.17487/RFC8785 . https://www.rfc-editor.org/info/rfc8785

Schuirmann, D.J.: A comparison of the two one-sided tests procedure and the power approach for assessing the equivalence of average bioavailability. Journal of Pharmacokinetics and Biopharmaceutics **15**(6), 657–680 (1987) https://doi.org/10.1007/BF01068419

Siebert, J., Joeckel, L., Heidrich, J., Trendowicz, A., Nakamichi, K., Ohashi, K., Namba, I., Yamamoto, R., Aoyama, M.: Construction of a quality model for machine learning systems. Software Quality Journal **30**(2), 307–335 (2022) https://doi.org/10.1007/s11219-021-09557-y

Singh, A.: Certifying compressed language models: An audit and a statistical toolkit. arXiv:2608.15046 (2026). https://doi.org/10.48550/arXiv.2608.15046 . https://arxiv.org/abs/2608.15046

Torres-Arias, S., Afzali, H., Kuppusamy, T.K., Curtmola, R., Cappos, J.: in-toto: Providing farm-to-table guarantees for bits and bytes. In: 28th USENIX Security Symposium (USENIX Security 19), pp. 1393–1410. USENIX Association, Santa Clara, CA (2019). https://www.usenix.org/conference/usenixsecurity19/presentation/torres-arias

Ville, J.: Étude Critique de la Notion de Collectif. Gauthier-Villars, Paris (1939)

Waudby-Smith, I., Ramdas, A.: Confidence sequences for sampling without replacement. In: Advances in Neural Information Processing Systems, vol. 33, pp. 20204–20214 (2020). https://proceedings.neurips.cc/paper/2020/hash/e96c7de8f6390b1e6c71556e4e0a4959-Abstract.html

Waudby-Smith, I., Ramdas, A.: Estimating means of bounded random variables by betting. Journal of the Royal Statistical Society Series B: Statistical Methodology **86**(1), 1–27 (2024) https://doi.org/10.1093/jrsssb/qkad009

Wei, F., Liu, L.: Pricing the risk of runtime compression: Anytime-valid admission

and a served-output law for compressed serving state. arXiv:2608.15810 (2026). https://doi.org/10.48550/arXiv.2608.15810 . https://arxiv.org/abs/2608.15810

Wei, F., Liu, L., Wang, Z., Wang, C.: WitCert: Sound runtime risk observability and gating for KV-cache quantization. arXiv:2607.28699 (2026). https://doi.org/10.48550/arXiv.2607.28699 . https://arxiv.org/abs/2607.28699

Wellek, S.: Testing Statistical Hypotheses of Equivalence and Noninferiority, 2nd edn. Chapman & Hall/CRC, Boca Raton, FL (2010). https://doi.org/10.1201/EBK1439808184

Xiao, G., Lin, J., Seznec, M., Wu, H., Demouth, J., Han, S.: SmoothQuant: Accurate and efficient post-training quantization for large language models. In: Proceedings of the 40th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 202, pp. 38087–38099 (2023). https://proceedings.mlr.press/v202/xiao23c.html

Yang, A., et al.: Qwen3 technical report. arXiv:2505.09388 (2025). https://doi.org/10.48550/arXiv.2505.09388 . https://arxiv.org/abs/2505.09388

Zhou, Z., Ye, Z., Chen, Z., Li, B., Liu, F.: CELEUS: Certifiable and efficient LLM evaluation via e-processes. arXiv:2606.20820 (2026). https://doi.org/10.48550/arXiv.2606.20820 . https://arxiv.org/abs/2606.20820

Supplementary Files

This is a list of supplementary files associated with this preprint. Click to download.

- OnlineResource1ArtifactandReproductionDetails.pdf
- OnlineResource2CodeDataandCertificateArtifact.zip