# Model Collapse Despite Ground-Truth Verification

**Shuo Li Liu** [1]

## Abstract

Exact verifier labels do not stop a sampled source from losing target support. A proof checker can accept every retained artifact under its declared manifest while leaving target modes that never reach the verifier untested. Lean answers one local question: did this artifact check in the declared environment? It checks the surviving artifacts; it cannot restore target support removed by generation or filtering. We formalize this support-loss form of collapse for proof-supervised LLMs. Lean acceptance is exact for a sampled formal artifact; it is not a claim that the artifact distribution equals the target population. A target-risk claim concerns tasks $x \sim P$ in a different space $\mathcal{X}$. We therefore separate artifact checks from target-task risk. A target-risk report must declare an adapter and audit it over target modes. Our first result shows the cost of missing support: if a target mode has positive mass but zero anchored source mass, two systems can have the same verifier statistics while their target risks differ by that mode's mass. The verifier has not failed; the denominator has changed. Our fixed-model bound charges residual mass, unsupported target mass, and, on supported modes, verified source loss, loss-class discrepancy, formalization error, and holdout uncertainty. For hypothesis testing, a verified aggregate becomes evidence about $P$ only after the report declares the tested population $P$, $H_0$, $H_1$, the test level $\eta$, and the upper-bound statistic $T_{\mathrm{audit}}$. The rejection rule is then explicit, for example $T_{\mathrm{audit}} \leq \epsilon$.

## 1. Introduction

Ground-truth verification, read as exact labels on sampled artifacts, can look like an antidote to recursive model collapse [4]: keep only artifacts the verifier accepts. The checked source can still miss target-positive modes. We use model collapse in the support sense: after recursive or verifier-filtered generation, the final source law may no longer reach target-positive modes. The verifier can certify the retained artifacts without detecting that loss of target support. Lean can reject a bad artifact it sees, but it cannot create an example from a mode the generator no longer proposes.

A pass-rate statistic therefore has a source denominator. In a Lean benchmark, the numerator is local: the checker accepts or rejects the submitted artifact under the declared Lean manifest. The denominator comes from the source law over generated or collected artifacts. A target-risk claim, however, concerns tasks drawn from the declared target population. Unless the report anchors $S = \tau_{\#}Q$ to $P$, names the covered modes, and states the transfer assumptions, the pass rate is a source statistic, not a target guarantee.

This distinction is easy to miss because the verifier provides exact artifact labels. If $V_{\mathrm{Lean}}(z, a) = 1$, then $a$ is accepted as a proof term for the formal statement encoded by $z$, relative to the trusted kernel and declared Lean manifest: version, imports, options, library state, and axiom/admission policy. The same locality applies to unit-test suites: once cases and environment are fixed, a pass means these cases passed here, not that all inputs or deployments are covered. Wet-lab protocols have the same locality once the report fixes QC, thresholds, calibration, batch structure, and no-call rules. For a sampled theorem, test case, well, or specimen, the verdict or endpoint label is protocol-local, not population-level. The population question remains: does the checked object represent the target population named in the risk claim? The verifier answers its local question; the report must connect the verified source to the target law.

Our starting point is a hypothesis-testing version of the domain-adaptation lesson: exact source labels do not identify target risk where the source has no support. The null is the high-risk claim $H_0 : \mathsf{R}_P(h) > \epsilon$; the alternative is $H_1 : \mathsf{R}_P(h) \leq \epsilon$ under the declared audit. The statistic is not the pooled pass rate. It is a *risk ledger* that records target mass, anchored source mass, loss-class discrepancy, semantic or endpoint-to-target gap, and uncertainty. It asks the report to state both "how many arti-

[1]Princeton University. Correspondence to: Shuo Li Liu <shuoliliu@princeton.edu>.

*Presented at the ICML 2026 Workshop on Hypothesis Testing*, Seoul, South Korea, 2026. Non-archival workshop paper.

facts passed" and "what population those artifacts represent." The stronger claim is deliberately more expensive: estimating these terms may require target sampling, cell annotation, adapter review, semantic adjudication, and protected conditional holdouts rather than a single verifier pass rate.

**Contributions.** We make four contributions. First, we give a typed setup for verifier-supervised tests with separate source and target spaces $\mathcal{Z}$ and $\mathcal{X}$, separate heads $h_Z$ and $h_X$, a declared adapter $\tau : \mathcal{Z} \to \mathcal{X}$, and a full target partition $C_0, \ldots, C_m$. Second, we prove an unsupported-mode result: if $q_j = 0 < p_j$, source-side verifier statistics cannot identify target behavior on $C_j$. Third, we state a fixed-model target-risk audit bound on independent holdouts. The bound prices supported cells by verified source loss, a loss-class discrepancy, a formalization or endpoint-gap term, and concentration; it prices unsupported cells by target mass. Fourth, we turn the theorem into a computable reporting protocol with support rules, simultaneous confidence accounting, semantic-audit categories, and a cellwise ledger table. For hypothesis testing, an exact artifact label enters a rejection rule for $P$ only after the denominator, adapter, and audit have been named; otherwise it remains evidence about $Q$. The primary partition is part of the estimand: too coarse a partition can hide missing submodes, while too fine a partition can make support operationally unavailable, so reports should predeclare the claim-relevant partition and include limited coarsening/refinement sensitivity.

This contract is conditional. It is not an impossibility theorem for proof benchmarks and not an empirical claim that current Lean datasets fail. It is a measurement rule for when verifier success can be used as evidence about a target law. If the source really represents the target and the adapter is audited, the ledger can be small. If a target mode is absent from the anchored source, the ledger prevents silent normalization.

**A running example.** Imagine a proof-supervised LLM benchmark that reports a $92\%$ Lean pass rate on a held-out set of formal artifacts. The number is meaningful as a statement about Lean acceptance under the declared manifest. The tempting leap is to read this as reliability on a target population of mathematical tasks. The target population may include routine algebra, induction arguments, geometry, combinatorics, translated natural-language proofs, and scientific lemmas. If the formal source places most mass on algebraic simplification and list induction, the $92\%$ is not wrong; it is a precise number over the wrong denominator for the broader claim. Target-heavy modes that were not formalized, sampled, or reachable through the adapter are not failed by Lean; they were never counted.

Verifier precision is most useful after the denominator is explicit. Once Lean removes source-label noise, the remaining question is coverage: which population did the verified source cover, and which target modes did it miss? The ledger preserves the value of proof checking while refusing to turn an off-support source statistic into a generalization certificate.

**What would count as success?** The ledger is fail-closed, not anti-verifier. It supports a strong claim when the target law is declared and the partition covers the intended modes. The anchored source must assign mass to the target-heavy cells; the loss-class discrepancy, formalization error, and final holdouts must also be audited. In that case, a low verifier loss becomes meaningful evidence about $P$. When a target mode is absent from the anchored source, the ledger makes the absence visible instead of normalizing it away.

## 2. Setup: What a Verified Aggregate Must Denote

Let $\mathcal{X}$ be the target task space with target law $P$, label map $y : \mathcal{X} \to \mathcal{Y}$, and bounded loss

$$L_X^h(x) = \ell_X(h_X(x), y(x)) \in [0, 1].$$

Write $\mathsf{R}_P(h) = \mathbb{E}_{X \sim P} L_X^h(X)$. Let $\mathcal{Z}$ be the verifier-artifact space with final source law $Q$. A system is typed as $h = (h_X, h_Z)$: the target head $h_X$ acts on $\mathcal{X}$, while the artifact head $h_Z : \mathcal{Z} \to \mathcal{A}$ produces an artifact for $z \in \mathcal{Z}$. The verifier $V_{\text{Lean}} : \mathcal{Z} \times \mathcal{A} \to \{0, 1\}$ induces the source loss

$$L_Z^h(z) = 1 - V_{\text{Lean}}(z, h_Z(z)) \in [0, 1].$$

Under the declared Lean manifest, $V_{\text{Lean}}(z, a) = 1$ means $a$ checks as a proof of the formal statement encoded by $z$. The manifest records the trusted kernel version, imports, options, library state, and axiom/admission policy. This verdict does not assert that $z$ was sampled from the target law, that every target cell has anchored source mass, that the formal statement captures the intended target label, or that $L_Z^h$ controls $L_X^h$.

The semantic bridge is a measurable adapter $\tau : \mathcal{Z} \to \mathcal{X}$. It induces the anchored source law

$$S = \tau_{\#} Q.$$

If the protocol allows abstention rather than an adapter value, the abstention rule must be predeclared; source abstention mass is reported under $Q$, and any target mass left unreachable after abstention is charged in the full-target certificate. The audit also fixes, before sampling, a full measurable target partition

$$\mathcal{C} = \{C_0, C_1, \ldots, C_m\}.$$

**Typed verifier-to-target chain.**

$$\begin{array}{ccccc} Z \sim Q & \xrightarrow{h_Z} & \text{formal artifact} & \xrightarrow{V_{\text{Lean}}} & L_Z^h \\ \tau \downarrow & & & & \downarrow \text{source statistic} \\ X \sim S = \tau_\# Q & \xleftrightarrow{d_{\mathcal{H}}(P_j, S_j)} & X \sim P & \xrightarrow{h_X, y} & L_X^h. \end{array}$$

A target-risk claim must state $P, C_0, \ldots, C_m, p_j, q_j, \tau$, the discrepancy class, and the formalization audit. The verifier checks the top row; the benchmark claim lives on the bottom row.

*Figure 1.* The verifier checks the artifact on the top row. Any claim about $P$ on the bottom row needs a separate audited source-to-target bridge.

The cell $C_0$ is the declared residual or outside-cover region. The named cells $C_1, \ldots, C_m$ are the target modes whose risk the report may claim. Define raw masses

$$p_j = P(C_j), \qquad q_j = S(C_j).$$

We use two support notions. Population support means $q_j = S(C_j) > 0$; operational support means the finite audit has both a positive lower confidence bound and enough conditional examples. The raw $p_j, q_j$ masses must stay visible. A cover-normalized score over supported cells is a useful secondary summary, but it is not a full-target claim unless $p_0 = 0$ and every named target cell has population and operational source support.

For $j \geq 1$ with $p_j > 0$ and $q_j > 0$, define $P_j = P(\cdot \mid C_j)$, $Q_j = Q(\cdot \mid \tau(Z) \in C_j)$, and $S_j = \tau_\# Q_j$. The loss-class discrepancy is the integral probability metric

$$d_{\mathcal{H}}(P_j, S_j) = \sup_{h \in \mathcal{H}} \left| \mathbb{E}_{P_j} L_X^h(X) - \mathbb{E}_{S_j} L_X^h(X) \right|. \quad (1)$$

Use MMD for this term only when the chosen kernel class controls the induced bounded loss class; otherwise it is a two-sample diagnostic rather than a labeled-loss certificate [2; 11; 12].

The one-sided formalization gap is separate from the discrepancy. For a fixed evaluated $h$, define

$$\epsilon_{\text{form},j}(h) = \mathbb{E}_{Z \sim Q_j} \left[ \left( L_X^h(\tau(Z)) - L_Z^h(Z) \right)_+ \right]. \quad (2)$$

This term charges semantic mismatch, not kernel unsoundness: missing assumptions, a wrong theorem statement, an adapter that changes the task, or a formal proof of a true statement that is not the statement the target task required. Lean soundness alone gives no upper bound on $\epsilon_{\text{form},j}(h)$.

The ledger keeps four denominators apart. The source-weighted verifier loss, $\mathbb{E}_Q L_Z^h(Z)$, is what a Lean pass rate measures most directly. Its cellwise version uses $r_j(h) = \mathbb{E}_{Q_j} L_Z^h(Z)$. The target-weighted supported term uses $p_j$, not $q_j$, and only for cells with $q_j > 0$ and certified finite-audit support. A cover-normalized score reports performance after the report has explicitly narrowed the claim to the covered subpopulation. Full target risk keeps the original denominator $P$, including $C_0$ and unsupported named modes. Each reported number should carry one of these labels, so readers do not mistake a source statistic for a full-population claim.

| Reported number | What must be true to read it beyond the source sample |
|---|---|
| Pooled Lean pass rate | It is a source-space statistic until the induced law $S = \tau_\# Q$, adapter, and coverage are audited against $P$. |
| Target-weighted supported certificate | Requires population and operational support, discrepancy, formalization, and holdout uncertainty for each supported cell. |
| Cover-normalized score | Must state the excluded target mass and cannot be used as full-target risk. |
| Full target-risk claim | Must charge $p_0$, unsupported named mass, transfer gaps, formalization gaps, and uncertainty. |

To evaluate a benchmark, the required objects are concrete: the target law $P$, the cell definition, the source process for $Q$, the adapter $\tau$, cells with $p_j > 0 = q_j$, the discrepancy class, and the audited formalization errors. A narrow benchmark may declare them briefly and a broad benchmark may need more detail, but these objects should be explicit before the pass rate is interpreted.

## 3. Unsupported Modes Are Not Identified

The failure mode is model collapse stated as a testing problem: the generated-and-verified source can lose a target mode, and after that loss the verifier can keep saying "valid" on the surviving modes. It is the same off-support uncertainty that appears in dataset shift, now with the label noise removed.

**Proposition 1** (Unsupported-mode non-identifiability)**.** *Fix the source law $Q$, adapter $\tau$, and a target cell $C_j$ with $p_j > 0$ and $q_j = 0$. Use 0-1 target loss. Suppose $\mathcal{H}$ contains two systems $h^0, h^1$ whose artifact heads and verifier losses agree $Q$-almost surely on $\mathcal{Z}$, and whose target heads agree on $\mathcal{X} \setminus C_j$, with $h^0$ correct on $C_j$ and $h^1$ wrong on $C_j$. Then any statistic whose inputs are restricted to source draws $Z \sim Q$, their verifier outcomes, and measurable post-processing of those quantities, with no target samples and no target-head queries, has the same distribution under $h^0$ and $h^1$. Their target risks differ by $p_j$.*

Proposition 1 makes a minimal claim. It does not say that Lean causes target failure. It says that a source-side aggregate, no matter how cleanly verified, cannot identify behavior in a target-positive region that the anchored source never reaches. Without a separate structural or semantic coverage

assumption, the audit cannot certify that cell; it should report the missing mass and decline the unsupported target claim. The denominator problem is that the pass rate is a precise number over $Q$, but $S = \tau_{\#}Q$ assigns no mass to $C_j$.

The proposition is best read as an indistinguishable-world argument. In one world, the model handles the missing target mode; in another, it fails that mode. The source artifacts and verifier outcomes are identical because the source never visits the mode. No amount of additional source sampling changes the conclusion while $q_j = 0$. Equivalently, a model can collapse even when every retained example has an exact local label: the verifier checks the surviving artifacts, not target mass lost before checking.

Near-collapse regimes are the finite-sample version of the same warning. If $0 < q_j \ll p_j$, then a passive source audit may draw too few conditional examples to estimate $L_Z^h$, $d_{\mathcal{H}}$, or $\epsilon_{\mathrm{form},j}$ with useful precision. A tiny $q_j$ counts as support only if the protocol can draw the planned conditional sample and attach a simultaneous uncertainty bound. Verification creates precise labels for sampled artifacts. It does not create probability mass.

## 4. The Fixed-Model Risk Ledger

Let

$$J = \{j \in \{1, \dots, m\} : p_j q_j > 0\},$$
$$\alpha_{\mathrm{unsup}} = \sum_{j \geq 1 : q_j = 0} p_j.$$

The residual outside-cover term is $p_0$. If $J = \emptyset$, interpret the population-supported-cell sum below as zero and omit the radius term. For each final-audit supported cell $j \in J$, draw an independent stratified holdout sample $Z_{j1}, \dots, Z_{jn_j} \sim Q_j$, independent of training, cover construction, prompt design, model selection, and formalization-rubric fitting. Define the empirical verifier loss

$$\widehat{r}_j(h) = \frac{1}{n_j} \sum_{k=1}^{n_j} L_Z^h(Z_{jk}).$$

**Theorem 1** (Full-target fixed-model audit bound)**.** *Fix $\eta \in (0,1)$, $h \in \mathcal{H}$, $P, Q, \tau$, the partition, the losses, the discrepancy class in Equation (1), and the sampling protocol before final audit data are inspected. Assume the evaluated loss function $x \mapsto L_X^h(x)$ belongs to that discrepancy class and all losses lie in $[0,1]$. Then, with probability at least $1 - \eta$ over the independent stratified verifier holdouts,*

$$\mathsf{R}_P(h) \leq p_0 + \alpha_{\mathrm{unsup}} + \sum_{j \in J} p_j \Bigg[ \widehat{r}_j(h) + d_{\mathcal{H}}(P_j, S_j) + \epsilon_{\mathrm{form},j}(h) + \sqrt{\frac{\log(|J|/\eta)}{2n_j}} \Bigg]. \tag{3}$$

*If a finite menu $\mathcal{H}_0$ is fixed before the final audit split and the reported $h \in \mathcal{H}_0$ is chosen after inspecting only these verifier-holdout losses, replace $\log(|J|/\eta)$ in the radius by $\log(|J||\mathcal{H}_0|/\eta)$. Broader adaptivity requires sample splitting or a separate uniform-convergence argument.*

This probability statement covers only the final verifier-holdout draw, with $P, Q, \tau, \mathcal{C}$, the discrepancy class, and the population gaps $\{\epsilon_{\mathrm{form},j}(h)\}_{j \in J}$ fixed. If $p_j$, $q_j$, $d_{\mathcal{H}}(P_j, S_j)$, or $\epsilon_{\mathrm{form},j}(h)$ are estimated, their confidence bounds must enter through the simultaneous certificate in the appendix.

Read the bound as an accounting rule for the benchmark claim. The $p_0$ term charges target mass outside the named cover. The $\alpha_{\mathrm{unsup}}$ term charges target-positive cells that the anchored source never samples. In supported cells, $d_{\mathcal{H}}(P_j, S_j)$ accounts for moving from the target conditional to the anchored-source conditional. The formalization term covers cases where the checked Lean statement is not the target task. The radius covers finite holdout data.

It is often useful to compare the target-weighted verifier term with the supported-cell source-weighted verifier loss. If a report gives

$$\widehat{R}^q_{\mathrm{Lean}}(h) = \sum_{j \in J} q_j \widehat{r}_j(h),$$

then for $r_j \in [0,1]$,

$$\sum_{j \in J} p_j r_j \leq \sum_{j \in J} q_j r_j + \gamma_{\mathcal{C}}(P, S),$$
$$\gamma_{\mathcal{C}}(P, S) = \sum_{j \in J} (p_j - q_j)_+.$$

$\gamma_{\mathcal{C}}$ is only a supported-cell reweighting diagnostic; it does not include $p_0$ or $\alpha_{\mathrm{unsup}}$, and it is not a substitute for the full ledger. The penalty is not a nuisance constant. It names a different estimand. A source-weighted pass rate can improve by oversampling easy formal modes while target-weighted risk worsens on target-heavy modes. The ledger makes the denominator explicit.

A strong verified benchmark makes every term small: source artifacts pass, anchored and target conditionals match for the claimed loss class, the audited formalization gap is small for the intended task, and the final audit has

$$\underbrace{p_0}_{\text{outside cover}} + \underbrace{\alpha_{\text{unsup}}}_{\text{unsupported modes}} + \sum_{j \in J} p_j \left[ \underbrace{\widehat{r}_j}_{\text{verified source loss}} + \underbrace{d_{\mathcal{H}}}_{\text{loss transfer}} + \underbrace{\epsilon_{\text{form},j}}_{\text{semantic/formalization gap}} + \underbrace{\rho_j}_{\text{finite sample}} \right].$$

**Fail-closed interpretation.** A missing term is not silently set to zero. If the report lacks a support estimate, discrepancy audit, formalization audit, or independent holdout radius, the corresponding target-risk claim remains unpriced.

*Figure 2.* The risk ledger corresponding to Equation (3). Lean supplies the verified source-loss term; the target claim also needs coverage, transfer, semantics, and uncertainty.

| Cell | $\hat{p}_j$ | $\hat{q}_j$ | Status | $\widehat{r}_j$ | $d_j^+$ | $\epsilon^+_{\text{form},j}$ | Claim |
|---|---|---|---|---|---|---|---|
| Arithmetic mode 1 | .55 | .70 | supported | .04 | .03 | .02 | priced |
| Arithmetic mode 2 | .40 | .30 | supported | .12 | .08 | .04 | priced |
| Residual $C_0$ | .05 | – | outside cover | – | – | – | charged |

*Table 1.* Illustrative ledger arithmetic, not an empirical benchmark result. With $n_1 = n_2 = 200$ and $\eta = .05$, the simultaneous holdout radius is about $0.096$, yielding a conservative certificate near $0.29$ while the pooled source verifier loss is $0.064$.

enough independent conditional data. Only then does the pass rate become credible evidence about $P$.

The bound is not meant to be tight in every application. A conservative upper bound can be loose and still be scientifically valuable if it identifies the unresolved risk. A large certificate dominated by $\alpha_{\text{unsup}}$ says the benchmark has a coverage problem. One dominated by $d_{\mathcal{H}}$ says the source and target conditionals are too far apart for the chosen loss class. One dominated by $\epsilon_{\text{form},j}$ says the formalization pipeline is the weak link. One dominated by the radius says the audit needs more independent conditional data. These are different engineering failures, and a single verifier pass rate cannot distinguish them.

**Corollary 1** (Zero verified loss is denominator-sensitive)**.** *If $\widehat{r}_j(h) = 0$ for every supported cell, Equation (3) still contains $p_0 + \alpha_{\text{unsup}}$, the discrepancy terms, the formalization terms, and the finite-sample radii. Thus zero verifier loss on the source supports a full target-risk claim only under additional coverage, transfer, formalization, and uncertainty conditions.*

The corollary states the core consequence. Perfect source labels remove source-label noise; they do not identify what the source never samples and what the adapter never validates.

## 5. Computable Certificate and Reporting Protocol

The population terms in Theorem 1 become useful only when every term has a reporting rule. The audit estimates $p_j$ from an iid target sample with mode labels and $q_j$ from an anchored-source mode sample or documented sampling frame. A cell is operationally supported only when a simultaneous lower confidence bound for $q_j$ is positive and the prespecified conditional count can be drawn. If the protocol cannot draw that count, the finite-audit certificate charges the cell's target mass as operationally unsupported; $\alpha_{\text{unsup}}$ remains the population zero-support term.

For a finite hypothesis class $\mathcal{H}_0$, the empirical loss-class IPM is the supremum over $h \in \mathcal{H}_0$ of the difference between target and anchored-source holdout averages of $L_X^h$, with simultaneous Hoeffding radii. For infinite classes, the report must state the VC, Rademacher, or kernel-IPM assumptions that justify the empirical process term. Domain-adversarial or distributionally robust methods are relevant here as modeling tools, but they do not remove the need to report support and uncertainty [7; 9; 10].

The formalization audit is separate. It samples anchored source items within supported cells and compares three objects: the informal target item, the anchored target object $\tau(z)$, and the submitted Lean statement. The protocol uses two independent annotations, adjudicates disagreements, and unblinds model losses only afterward. Rubric categories include correct formalization, missing assumption, changed conclusion, wrong target, ambiguous adapter, dependency mismatch, and label/task mismatch. When categories map to bounded loss penalties, cellwise Clopper–Pearson, Wilson, or empirical-Bernstein intervals provide simultaneous upper bounds.

Table 1 is a reporting template, not evidence about any real Lean dataset. Its purpose is to prevent the common substitution of a source-weighted pass rate for a target-risk audit. The same table should be produced for any proof-supervised benchmark that wants to claim target-task validity: each row names a cell, its target mass, its anchored source mass, its support status, its verified loss, its discrepancy upper bound, its formalization-gap upper bound, and its contribution to the final certificate.

## 6. Evaluation Science Without New Empirical Overclaiming

Without a live audit, evaluation should specify what future audits must measure before a verifier aggregate becomes target-risk evidence. For benchmark reports seeking target-risk validity, four diagnostic stress tests are natural when data permit.

First, an uncovered-mass stress test varies whether predeclared target cells appear in the source. It holds source size, model class, and held-out verifier loss fixed by design, or reports the remaining mismatch. Its estimand is the target-loss contrast on a target holdout for the omitted cells. A positive contrast flags unsupported mass; a null result only says those cells did not drive this design. Second, a mode-

weight stress test keeps support fixed but changes $q_j$, comparing source-weighted and target-weighted reports. Third, a within-mode transfer test adds target-near variants rather than duplicates of an existing formal pattern, then asks whether $d_{\mathcal{H}}(P_j, S_j)$ or cell target loss decreases. Fourth, a curriculum audit tracks $Q_0, \ldots, Q_T$ as a diagnostic path unless a separate sequential theorem is stated. Curriculum learning can matter [3], but the fixed-model theorem uses only the final audited law $Q_T$.

These tests should be read beside modern benchmark, shift, and model-collapse practice. WILDS-style distribution-shift evaluation makes domain and subgroup structure explicit [8]; proof-supervised systems such as LeanDojo, process-verified theorem-proving, ProofBridge, and SorryDB make formal artifacts increasingly central to LLM evaluation [16–19]. The ledger states the conditions under which a proof-checked source statistic may be read as target evidence. It says that a proof-checked label set is a strong source-space object, but target validity still depends on the declared population frame and the support of the adapter.

A conventional results paragraph might stop at "the model achieves a high Lean pass rate on the held-out benchmark." A target-risk result names the covered target mass, unsupported mass, discrepancy and formalization bounds, and independent audit split. The fuller result also tells the next experiment where to look: the largest ledger term.

In hypothesis-testing terms, the object under test is the target-risk reading of a verified aggregate. The ledger defines an upper-bound statistic and rejection rule: at test level $\eta$, reject the high-risk null only when the full certificate, with unsupported mass charged, is at most the declared tolerance $\epsilon$. Many LLM evaluations now combine exact artifact checks with unstable source distributions, so the target-risk reading has to be tested rather than inferred from the pass rate. Recursive generation can shrink the support of what models see; ground-truth verification can then make the surviving support look unusually clean. Formal traces, proof certificates, retrieval corpora, and verifier-trained policies can make a source result look complete. A clean artifact verdict is valuable; it is not a verdict about the target population.

## 7. What Is New, and What Is Not

The novelty is the typed contract that a verified benchmark must satisfy before a verifier aggregate becomes a target-risk claim, rather than a better proof checker or a sharper generic adaptation theorem. In ordinary adaptation notation it is tempting to write down a source distribution and a target distribution and then invoke a discrepancy. In proof-supervised evaluation this misses two important typing facts. First, the source loss is a verifier loss on formal artifacts, while the target loss is a task loss on target items. Second, the bridge between those losses is not automatic; it is the adapter and formalization audit. The theorem is a specialization, but one that changes the required reporting obligations.

The ledger also changes the role of examples. A single formal artifact can be perfectly checked and still be irrelevant to the target distribution. A large library can be valuable and still underweight a target-heavy mode. A target item can be faithfully formalized, or it can be formalized into a nearby theorem with different assumptions. These are not philosophical objections; they are terms in Equation (3). By writing them as terms, the ledger turns a vague worry about "benchmark validity" into quantities that can be audited, bounded, or explicitly left unresolved.

No live ledger audit on an existing proof-supervised benchmark is claimed here. The contribution is the certificate form, theorem, reporting table, and audit protocol needed to make such a study interpretable. A benchmark supports a target-risk reading only after each ledger term is measured or explicitly charged. The reporting rule preserves the value of verification while making clear what verification has identified.

## 8. Related Work

**Domain adaptation and dataset shift.** The theorem specializes a familiar template: target risk is controlled by source risk plus a discrepancy term, a label or joint-error term, support conditions, and concentration [1; 5; 6]. Our contribution is not to rediscover these ingredients, but to type them correctly for verifier-certified supervision. In this setting the source loss is unusually well measured, but the population frame, adapter, unsupported mass, and formalization error are still external obligations.

**Discrepancy estimation.** Integral probability metrics and kernel two-sample methods give tools for measuring source-target differences [2; 11; 12]. The ledger uses those tools only with their assumptions attached. An unlabeled two-sample statistic is a diagnostic unless it controls the relevant bounded loss class; MMD is admissible only when the chosen kernel class controls the induced losses. This distinction is important for proof benchmarks, where a syntactic or embedding-level similarity score can look reassuring while target loss remains unpriced.

**Formal verification and proof-supervised LLMs.** Lean, mathlib, and adjacent Lean libraries provide precise formal infrastructure [13–15]. LLM theorem-proving systems then use formal proof states, retrieval, and process verification as supervision or evaluation [16; 17; 19; 20]. These systems make verifier labels more valuable, but

also make the denominator more important: the source of formal artifacts, their target interpretation, and the unsupported modes must be reported.

**Uncertainty and reporting.** The ledger is also a reporting discipline. When audits use clustered data, repeated theorem families, or shared formalization sources, uncertainty should respect that dependence rather than treating every artifact as independent [21; 22]. The main theorem states the simplest independent-holdout version; the appendix records how the certificate changes under simultaneous confidence accounting and declared dependence structure.

## 9. Failure Modes and Repairs

The ledger is useful because its failure modes suggest repairs. If $\alpha_{\mathrm{unsup}}$ dominates the certificate, the problem is not the verifier; it is the source frame or adapter reachability. The repair is to collect or formalize examples in the missing target cells, or to narrow the benchmark claim to the covered population. If $\gamma_{\mathcal{C}}$ is large while every named cell is supported, the source is present but misweighted. The repair is target-weighted reporting, rebalancing, or a design that explicitly states that the claim is source-weighted. If $d_{\mathcal{H}}$ is large inside a supported cell, then the examples in that cell are not representative for the evaluated loss class. The repair is not merely more examples; it is more target-near examples or a different adapter.

If $\epsilon_{\mathrm{form},j}$ dominates, the bottleneck is semantic. The formal artifacts may be valid Lean objects but poor formalizations of the intended target tasks. This is where formal-methods expertise matters most. A better theorem statement, clearer assumptions, or a narrower target rubric can reduce the gap. If the concentration radii dominate, the audit is underpowered. The repair is more independent conditional data, cluster-aware sampling, or a more modest claim. These repairs are different, and a pooled pass rate does not tell the benchmark builder which one is needed.

The fail-closed stance also protects positive claims. A benchmark that reports small unsupported mass, small discrepancy, low formalization error, and honest uncertainty has a stronger story than one that reports only a high pass rate. The former tells readers why the verified labels should transfer. The latter asks readers to infer a target denominator that may not exist. Proof checking is an excellent measurement primitive, but measurement primitives need population frames.

## 10. Using the Ledger in Hypothesis Testing

In a hypothesis-testing report, the target-risk claim should replace the pooled pass rate. Open with that claim, then show the evidence rows: target population, target and anchored-source denominators $p_j, q_j$, verifier loss, discrepancy bound, formalization-gap bound, uncertainty, and a claimed, charged, or unresolved flag. This is the minimum information needed to tell a source statistic from a certificate about $P$.

The same discipline applies to abstracts: verified labels remove source-label noise, target validity is claimed only for audited covered cells, and unsupported target mass is charged separately. This respects the proof assistant by not asking it to certify distributional facts it never observed.

The ledger also supports negative or partial results. If only half of the intended target mass can be audited, the claim becomes narrower: a certificate for the covered half of $P$ and an explicit open problem for the rest. A large formalization-gap interval is likewise useful if it identifies where formalization tooling, annotation rubrics, or benchmark design should improve.

Finally, the ledger is compatible with progress. If a new autoformalization method expands support, the certificate should show $\alpha_{\mathrm{unsup}}$ shrinking. If a retrieval method improves target-near examples, the discrepancy term should shrink. If a proof-state curriculum improves source proficiency but leaves $p_j, q_j$ unchanged, the verifier-loss term may shrink while the support terms do not. These distinctions let the field tell different kinds of improvement apart. The practical payoff is a verified evaluation that reports how formal evidence supports target claims alongside any leaderboard number.

## 11. Testing Validity Rather Than Only Accuracy

The Hypothesis Testing framing is direct. We use the strict high-risk null $H_0 : \mathsf{R}_P(h) > \epsilon$ and the weak target-risk alternative $H_1 : \mathsf{R}_P(h) \leq \epsilon$, with equality included in the alternative. The statistic is the simultaneous ledger upper bound $T_{\mathrm{audit}}(h)$, the finite-audit version of the right-hand side of Equation (3), not the pooled pass rate. The rejection region is $T_{\mathrm{audit}}(h) \leq \epsilon$, after the evaluated $h$, $P$, $\tau$, $\mathcal{C}$, losses, splits, audits, and the test level $\eta$ are fixed before final audit data are inspected. The simultaneous audit event covers every estimated upper bound and support declaration used to form $T_{\mathrm{audit}}$. On that event, $\mathsf{R}_P(h) \leq T_{\mathrm{audit}}(h)$. Thus any rejection under $H_0 : \mathsf{R}_P(h) > \epsilon$ occurs only off the event, and the Type-I error is at most $\eta$ under the stated assumptions. Source pass rates enter only through supported cells. Unsupported cells are charged rather than treated as missing observations.

This view also explains why exact local labels do not trivialize the testing problem. In ordinary supervised testing, uncertainty comes from labels, sampling, model selection,

and distribution shift. Lean and unit tests can supply exact labels for the checked artifact or case; wet-lab assays supply protocol-qualified endpoint labels. Neither removes uncertainty about whether the source sample represents the target population. The unsupported-mode lower bound is therefore not a pessimistic add-on; it is a diagnostic showing that a source-only statistic cannot support the stated target-risk test. If a target-positive cell has zero anchored source support, then no source-only test statistic can distinguish two systems that differ only on that cell.

The practical testing protocol is consequently cellwise. Define the target claim before final audit data. Assign target items to the predeclared partition, estimate raw target masses, and push source artifacts through $\tau$ to estimate anchored source masses. Mark unsupported or low-count cells before computing any pooled pass rate. For supported cells, bound verifier loss, discrepancy, formalization gap, and finite-sample uncertainty separately. The full certificate reports a target-risk reading only when the charged ledger total is below the declared tolerance under the stated assumptions.

Compared with a leaderboard pass rate, the ledger gives more diagnostic information. A failed ledger identifies which link in the claim is unresolved. The prover may fail on represented artifacts, the source may miss target-heavy cells, the adapter may point to the wrong informal task, the within-cell distributions may differ, or the final audit may be too small. Each unresolved term points to a different repair.

For readers, the result is a sharper claim. They do not have to infer whether "verified accuracy" means a source statistic, a supported-cover statistic, or a full-population statement. A clear report puts the denominator on the first page and shows unsupported mass before reporting the pass rate. That is the central testing contribution: verified labels make the source outcome clean enough that the remaining validity assumptions can be tested, priced, or rejected.

## 12. Limitations and Conclusion

The scope is narrow by design. The guarantee is for fixed models on independent audit holdouts. It assumes a declared target distribution, a measurable adapter, a predeclared target partition, bounded or controlled losses, an estimable discrepancy class, and an auditable formalization gap. If the model, prompts, adapter, partition, discrepancy class, or rubric are chosen after final audit data are inspected, the fixed-model reading no longer applies. The report must then pay for selection with a finite-class correction, sample splitting, or a separate uniform theorem.

The claim is not that existing proof benchmarks fail or that verification is weak. Exact verification removes source-label noise, leaving the denominator question exposed. A checked artifact is exact evidence about a formal statement; an assay can supply protocol-local evidence about a measured endpoint. What remains is the denominator: which target distribution does that statement or endpoint represent, if any? When a target-positive mode is off support, collapse can persist: the verifier checks the survivors, not the missing mass. Where the mode is supported and the adapter, discrepancy, formalization, and uncertainty terms are small, the ledger explains why verifier-trained evaluation can be meaningful.

The intended use is modest and practical. A reader should be able to locate the target population, cell masses, support flags, adapter, discrepancy class, formalization audit, and uncertainty budget without reconstructing the benchmark from prose. If these objects are absent, the pass rate may still be a useful source diagnostic, but not a test of target validity.

The positive message for Hypothesis Testing is that strong tests say what evidence would change the conclusion. The ledger says exactly how a proof-supervised benchmark can improve: cover the missing target cells, rebalance source and target weights, audit the semantic bridge, estimate the loss discrepancy, and protect the final split from adaptation. Those repairs turn a clean but local verifier statistic into a defensible target-risk claim.

For a reviewer, the checklist reduces to five questions. What are $H_0$ and $H_1$? What are the tested population $P$, the test level, the statistic, and the rejection region? Which target modes receive positive mass but no anchored source support? Which assumptions turn formal correctness on $\mathcal{Z}$ into task correctness on $\mathcal{X}$? Which data were held out until the final certificate was computed? If a benchmark report answers these questions, then its verified labels are tied to a defined target-risk claim. If it does not, the pass rate is still real, but it is evidence for the checked source process, not for the whole target population.

The checklist also prevents a common rhetorical slide. Authors may say that the benchmark has ground truth because the verifier is exact. The ledger asks: ground truth for which random variable, under which protocol? In the source space, the answer is clear. In the target space, the answer depends on support, the adapter, and the formalization audit. That distinction is the main contribution to testing methodology. It gives readers a way to accept strong verified evidence without accepting untested distributional conclusions.

## References


[1] S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. W. Vaughan. A theory of learning from different domains. *Machine Learning*, 79:151–175, 2010.

[2] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Scholkopf, and A. Smola. A kernel two-sample test. *Journal of Machine Learning Research*, 13:723–773, 2012.

[3] Y. Bengio, J. Louradour, R. Collobert, and J. Weston. Curriculum learning. In *Proceedings of the 26th International Conference on Machine Learning*, pages 41–48, 2009.

[4] I. Shumailov, Z. Shumaylov, Y. Zhao, N. Papernot, R. Anderson, and Y. Gal. AI models collapse when trained on recursively generated data. *Nature*, 631:755–759, 2024.

[5] J. Quinonero-Candela, M. Sugiyama, A. Schwaighofer, and N. D. Lawrence, editors. *Dataset Shift in Machine Learning*. MIT Press, 2009.

[6] M. Sugiyama, M. Krauledat, and K.-R. Mueller. Covariate shift adaptation by importance weighted cross validation. *Journal of Machine Learning Research*, 8:985–1005, 2007.

[7] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky. Domain-adversarial training of neural networks. *Journal of Machine Learning Research*, 17(59):1–35, 2016.

[8] P. W. Koh et al. WILDS: A benchmark of in-the-wild distribution shifts. In *Proceedings of ICML*, pages 5637–5664, 2021.

[9] S. Kim, S. H. Jo, W. Ha, and M. Chae. Distributionally robust classification for multi-source unsupervised domain adaptation. In *International Conference on Learning Representations*, 2026. https://openreview.net/forum?id=RyAyH8ufGu.

[10] S. I. Amoukou, E. Albini, T. Bewley, S. Mishra, and M. Veloso. Entropic projection alignment: Estimating, explaining, and improving model performance under distribution shift. In *AISTATS*, 2026. https://openreview.net/forum?id=6PuvcdqLzT.

[11] G. Lee, S. Shekhar, and I. Kim. A semi-supervised kernel two-sample test. In *AISTATS*, 2026. https://openreview.net/forum?id=FIp15n9OnW.

[12] B. T. Tran and N. Schreuder. Minimax-optimal two-sample test with sliced Wasserstein. In *AISTATS*, 2026. https://openreview.net/forum?id=1x1kCzeMyx.

[13] The mathlib Community. The Lean mathematical library. In *Proceedings of CPP*, pages 367–381, 2020. doi:10.1145/3372885.3373824.

[14] L. de Moura and S. Ullrich. The Lean 4 theorem prover and programming language. In *Automated Deduction – CADE 28*, pages 625–635, 2021. doi:10.1007/978-3-030-79876-5_37.

[15] C. Barrett, S. Chaudhuri, F. Montesi, J. Grundy, P. Kohli, L. de Moura, A. Rademaker, and S. Yingchareonthawornchai. CSLib: The Lean Computer Science Library. arXiv:2602.04846, 2026.

[16] P. Jana, K. Kale, A. E. Tanriverdi, C. Song, S. Vishwanath, and V. Ganesh. ProofBridge: Auto-formalization of natural language proofs in Lean via joint embeddings. In *International Conference on Learning Representations*, 2026. https://openreview.net/forum?id=U2jxHXuOX9.

[17] M. Kim and S.-Y. Yun. Process-verified reinforcement learning for theorem proving via Lean. In *International Conference on Learning Representations*, 2026. https://openreview.net/forum?id=P00k4DFaXF.

[18] A. Letson et al. SorryDB: Can AI provers complete real-world Lean theorems? arXiv:2603.02668, 2026.

[19] K. Yang, A. Swope, A. Gu, R. Chalamala, P. Song, S. Yu, S. Godil, R. Prenger, and A. Anandkumar. LeanDojo: Theorem proving with retrieval-augmented language models. In *Advances in Neural Information Processing Systems*, 2023.

[20] S. Polu and I. Sutskever. Generative language modeling for automated theorem proving. arXiv:2009.03393, 2020.

[21] A. C. Cameron and D. L. Miller. A practitioner's guide to cluster-robust inference. *Journal of Human Resources*, 50(2):317–372, 2015. doi:10.3368/jhr.50.2.317.

[22] A. Abadie, S. Athey, G. W. Imbens, and J. M. Wooldridge. When should you adjust standard errors for clustering? *Quarterly Journal of Economics*, 138(1):1–35, 2023. doi:10.1093/qje/qjac038.

# A. Full Notation and Estimands

The main text uses raw target masses because full-target risk is the estimand. Let $c : \mathcal{X} \to \{0, \dots, m\}$ be the mode map and $C_j = c^{-1}(j)$. Raw masses are $p_j = P(C_j)$ and $q_j = S(C_j)$. The residual $p_0$ is charged directly. The source residual $q_0$ is reported for source accounting, but it does not reduce target risk unless the target claim excludes $C_0$. Cover-normalized summaries

$$\bar{p}_j = \frac{p_j}{1 - p_0}, \qquad \bar{q}_j = \frac{q_j}{1 - q_0}$$

may be useful for descriptive comparison, but they must be labeled as supported-cover summaries. They are not substitutes for the raw-mass certificate in Equation (3).

For $j \in J$, the target conditional law $P_j$, source conditional law $Q_j$, and anchored conditional law $S_j = \tau_\# Q_j$ define the cellwise transfer problem. If $p_j = 0$, the cell contributes no target risk. If $q_j = 0 < p_j$, the cell contributes to $\alpha_{\text{unsup}}$. If $p_j q_j > 0$, the cell is eligible for the supported-cell bound, provided that the audit can draw the predeclared conditional count and complete the formalization audit.

## A.1. Ledger tuple

A complete audit record for a fixed model can be stored as the tuple

$$\mathsf{Ledger}(h) = \left(P, \mathcal{C}, \tau, \{p_j, q_j, s_j, \widehat{r}_j, d_j^+, \epsilon^+_{\text{form},j}, \rho_j\}_{j=0}^m, \eta, \mathsf{split}\right),$$

where $s_j$ records whether the cell is supported, population-unsupported, operationally unsupported, or outside cover; $d_j^+$ and $\epsilon^+_{\text{form},j}$ are upper confidence bounds, $\rho_j$ is the finite-sample radius, and split records how construction, selection, and final audit data were separated. A main-text table may be smaller, but the appendix should make this tuple reconstructible.

## A.2. Denominator declarations

Every aggregate should carry a denominator label. We use the following convention:

$$\begin{aligned}
\mathsf{src}(h) &= \mathbb{E}_Q L_Z^h(Z), \\
\mathsf{tgt\text{-}supp\text{-}term}(h) &= \textstyle\sum_{j \in J} p_j \mathbb{E}_{P_j} L_X^h(X), \\
\mathsf{cover\text{-}norm}(h) &= \left(\textstyle\sum_{j \in J} p_j\right)^{-1} \textstyle\sum_{j \in J} p_j \mathbb{E}_{P_j} L_X^h(X), \\
\mathsf{full}(h) &= \mathsf{R}_P(h).
\end{aligned}$$

The first line is the native verifier statistic. The second is the unnormalized supported-cell contribution; the third is its cover-normalized version. The fourth is the benchmark claim most readers infer unless the report says otherwise. The ledger is designed to prevent accidental movement between these four lines.

# B. Proof of Unsupported-Mode Non-Identifiability

Fix $C_j$ with $p_j > 0$ and $q_j = 0$. Since $q_j = S(C_j) = Q(\tau(Z) \in C_j)$, a source draw lands outside the anchored preimage of $C_j$ almost surely. By assumption, the artifact heads and verifier losses of $h^0$ and $h^1$ agree $Q$-almost surely on $\mathcal{Z}$. Therefore the verifier outputs, source losses, and any statistic measurable with respect to the source sample and verifier outcomes are identical in distribution under the two systems.

On the target side, the two systems agree outside $C_j$. Under 0-1 loss, $h^0$ is correct on $C_j$ and $h^1$ is wrong on $C_j$, so

$$\mathsf{R}_P(h^1) - \mathsf{R}_P(h^0) = P(C_j) = p_j.$$

Thus source-only verifier evidence cannot identify the target risk on a target-positive unsupported cell. The proof uses no weakness of Lean; it uses only the absence of source constraints on that cell.

# C. Proof of the Audit Bound

Write $\mathsf{R}_P(h) = \mathbb{E}_P L_X^h(X)$. Decompose over the partition:

$$\mathsf{R}_P(h) = \sum_{j=0}^m p_j \mathbb{E}_{P_j} L_X^h(X),$$

with the convention that zero-mass conditionals are omitted. Since $L_X^h \in [0, 1]$, the residual cell contributes at most $p_0$. Each named cell with $q_j = 0$ contributes at most $p_j$, giving the unsupported term $\alpha_{\text{unsup}}$.

For $j \in J$, the IPM definition gives

$$\mathbb{E}_{P_j} L_X^h(X) \le \mathbb{E}_{S_j} L_X^h(X) + d_{\mathcal{H}}(P_j, S_j).$$

Because $S_j = \tau_\# Q_j$,

$$\mathbb{E}_{S_j} L_X^h(X) = \mathbb{E}_{Q_j} L_X^h(\tau(Z)).$$

By the one-sided formalization gap,

$$\mathbb{E}_{Q_j} L_X^h(\tau(Z)) \le \mathbb{E}_{Q_j} L_Z^h(Z) + \epsilon_{\text{form},j}(h).$$

For the independent holdout in cell $j$, one-sided Hoeffding's inequality gives

$$\mathbb{E}_{Q_j} L_Z^h(Z) \le \widehat{r}_j(h) + \sqrt{\frac{\log(|J|/\eta)}{2n_j}}$$

simultaneously for all $j \in J$ with probability at least $1 - \eta$, by a union bound. Multiplying by $p_j$, summing over supported cells, and adding the residual and unsupported terms proves Equation (3). If a finite class $\mathcal{H}_0$ is selected after viewing holdout losses, the union bound also ranges over $\mathcal{H}_0$, giving $\log(|J||\mathcal{H}_0|/\eta)$. No broader adaptive claim is implied.

### C.1. Why the proof is not Lean-specific

The proof uses Lean only through the definition of $L_Z^h$ and the interpretation of $V_{\text{Lean}} = 1$. The inequality itself is a cellwise adaptation decomposition. That generality is useful. It means that the theorem isolates exactly where formal verification enters: it makes the source loss meaningful and auditable. The theorem does not ask Lean to solve support, target weighting, semantic mapping, or finite-sample uncertainty. Those quantities remain statistical.

### C.2. One-sided formalization gap

The main theorem uses a one-sided gap because an upper bound on target loss needs only the amount by which target loss can exceed verifier loss after applying the adapter. A symmetric absolute gap is also possible:

$$\tilde{\epsilon}_{\text{form},j}(h) = \mathbb{E}_{Q_j}\left|L_X^h(\tau(Z)) - L_Z^h(Z)\right|.$$

Using $\tilde{\epsilon}_{\text{form},j}$ gives a valid but sometimes more conservative bound. The one-sided version makes the audit target clearer: a formalization error matters for the upper bound when it lets a verifier success understate target loss. If the rubric cannot justify the direction, the symmetric version should be reported.

## D. Simultaneous Confidence Certificate

A computable certificate replaces every population term in Equation (3) by a simultaneous upper confidence bound or maximizes the right-hand side over a joint confidence set. Allocate a total failure budget $\eta$ across target weights, source weights, verifier losses, discrepancies, and formalization errors. For example, a simple Bonferroni allocation may assign $\eta/5$ to each family. Multinomial, Hoeffding, Wilson, or Clopper–Pearson intervals can be used for masses and bounded rates; empirical-Bernstein bounds may be used when variance estimates are part of the prespecified protocol.

A finite-audit support declaration should be conservative. The audit marks $C_j$ supported only if the lower confidence bound for $q_j$ is positive and the source sampling frame can deliver the required conditional count $n_j$. Otherwise the target mass is charged as operationally unsupported in the finite-audit certificate; $\alpha_{\text{unsup}}$ remains the population zero-support term. This rule is intentionally fail-closed. It is better for a certificate to be conservative than to convert a missing conditional sample into a misleading zero.

When target weights are uncertain, the final certificate can maximize

$$p_0 + \alpha_{\text{unsup}} + \sum_{j\in J} p_j\left[\widehat{r}_j^+ + d_j^+ + \epsilon_{\text{form},j}^+ + \rho_j\right]$$

over the joint confidence set for $p$. This is preferable to plugging in point estimates and then placing error bars only on the final number. The ledger should show which terms dominate the certificate, because that is the practical guidance for improving the benchmark.

### D.1. Audit algorithm

1. Declare $P$, the sampling frame for target tasks, and the mode map $c : \mathcal{X} \to \{0, \ldots, m\}$. Freeze the overlap and residual-cell rules.
2. Freeze the adapter $\tau$, the evaluated model $h$, the loss definitions, the discrepancy class, and the formalization rubric.
3. Draw independent target and source audit samples. Estimate $p_j$ and $q_j$, with simultaneous intervals.
4. Mark cells supported only if the support rule and conditional count rule both pass.
5. For supported cells, estimate verifier loss, discrepancy, and formalization error on samples not used to construct the model or rubric.
6. Compute the right-hand side of Equation (3) with all terms replaced by simultaneous upper bounds.
7. Report the cell table and global certificate, including failure codes for any unresolved term.

The algorithm has more steps than a single leaderboard score because the score hides the target-risk estimand. The audit exposes that estimand.

### D.2. Dependence and clustered artifacts

The main theorem assumes independent conditional holdouts. Real proof corpora may contain clusters: variants of a theorem, shared proof skeletons, generated families, or multiple artifacts derived from the same informal source. In such cases, the audit should either sample clusters independently and aggregate at the cluster level or use a cluster-aware confidence method. Treating many near-duplicates as independent artificially shrinks the radius. This issue is not special to Lean, but proof corpora can make it acute because syntactic variants are easy to generate.

## E. Formalization Audit Rubric

The formalization audit estimates the gap between target correctness and verifier correctness. It is not a second proof checker. It is an audit of whether the formal statement being checked is the statement that the target task intended.

| Category | Conservative interpretation |
|---|---|
| Correct formalization | No penalty beyond ordinary annotation uncertainty. |
| Missing assumption | Charge the sample unless the missing assumption is irrelevant under the target rubric. |
| Changed conclusion | Charge: the proved statement does not answer the target item. |
| Wrong object or type | Charge: the adapter maps to the wrong semantic object. |
| Ambiguous adapter | Charge or route to adjudication; do not silently set to correct. |
| Dependency mismatch | Charge if imported facts or library conventions change the declared target semantics. |
| Wrong label or task | Charge as formalization error, not verifier error. |

Two annotators should label each sampled item independently. Adjudication should happen before model losses are unblinded. If categories map to bounded penalties $b \in [0, 1]$, the cellwise formalization upper bound is an upper confidence bound on the mean penalty. If no such mapping is justified, the formalization audit remains qualitative and the target-risk certificate must state that $\epsilon_{\mathrm{form},j}$ is unaudited.

### E.1. Examples of charged errors

A natural-language task may ask for a theorem over positive integers, while the Lean statement proves a version over naturals with an extra nonzero condition. Whether this is harmless depends on the target rubric; the audit cannot assume it away. A geometry task may be formalized as an algebraic lemma that omits a nondegeneracy condition. A scientific-reasoning item may be converted into a theorem about an idealized model that no longer captures the target label. In all three cases the verifier can be perfectly sound while the target correspondence is partial.

The audit should also record benign differences. Not every syntactic mismatch is a target error. A formalization may strengthen a statement in a way that still answers the target task, or it may use a different library lemma without changing semantics. Adjudication separates harmless representation choices from changes that affect the loss claim. The important rule is procedural: decide this before seeing the model's final audit performance.

## F. Discrepancy, MMD, and Loss Classes

The theorem uses a labeled loss-class IPM. For a finite class $\mathcal{H}$, the empirical discrepancy in cell $j$ is

$$\widehat{d}_j = \sup_{h \in \mathcal{H}} \left| \frac{1}{n_{P,j}} \sum_{i:c(X_i)=j} L_X^h(X_i) - \frac{1}{n_{S,j}} \sum_{i:c(\tau(Z_i))=j} L_X^h(\tau(Z_i)) \right|$$

plus simultaneous concentration radii. For infinite classes, the report must state a VC, Rademacher, stability, or kernel-IPM argument. A kernel MMD is valid as the theorem's discrepancy only if the relevant loss functions are controlled by the chosen kernel class. Otherwise it is still useful as a diagnostic of distribution shift, but not as a bound on target loss.

This distinction matters in formal benchmarks because syntactic proximity can be misleading. Two Lean statements may look close under an embedding while differing in a hypothesis that changes the target problem. Conversely, two natural-language tasks may look different while mapping to the same formal theorem. The discrepancy term should therefore be tied to the loss claim being made, not merely to a representation convenience.

## G. Near-Collapse Support Scarcity

Near-collapse means finite-sample support scarcity. Suppose raw source auditing samples $N$ artifacts from $Q$. For a cell with $q_j > 0$, the count $N_j$ is binomial with mean $Nq_j$ under iid sampling. If $q_j$ is small, the audit may fail to obtain the conditional count needed by Theorem 1. The risk ledger should then either draw a stratified conditional sample from a documented frame or report the cell as operationally unsupported for the certificate.

This is not a claim that target error must be high in rare cells. It is a claim about what the audit can certify. Small $q_j$ inflates uncertainty and sampling cost. The proper benchmark response is to report the count, the confidence interval for $q_j$, the conditional losses if available, and the target mass that remains unpriced if the count is unavailable.

### G.1. A simple count calculation

Let $N_j \sim \mathrm{Binomial}(N, q_j)$ be the number of raw source-audit examples landing in cell $j$. If the certificate requires at least $n_{\min}$ conditional examples, then small $q_j$ forces large $N$. A crude but useful planning rule is that $Nq_j$ should be comfortably larger than $n_{\min}$ plus a concentration allowance. If this is not feasible, the benchmark should sample the cell conditionally from a documented source frame or report the cell as unsupported. What it should not do is report a precise pooled source pass rate and leave the rare target-heavy cell implicit.

This is the operational meaning of near-collapse for the ledger. The source may not have literally zero support, but the available audit design may still provide no reliable conditional estimate. The ledger's support flag is therefore an empirical declaration with uncertainty, not a metaphysical statement about the support of an ideal distribution.

## H. Adaptive Selection and Holdout Leakage

The theorem is a fixed-model statement. The final audit split should not be used to choose the model, prompt, checkpoint, adapter, partition, discrepancy class, or formalization rubric. A robust workflow uses three splits: a construction split for designing $\tau$, the partition, prompts, and rubrics; a selection split for choosing among systems and discrepancy classes; and a final audit split used once for the reported certificate.

The finite-class correction in Theorem 1 covers a limited case: selecting from a finite set of systems after seeing holdout losses. It does not cover interactive redesign of the benchmark. If final audit data influence the construction, the report should label the certificate exploratory or supply a uniform-convergence, stability, or reusable-holdout analysis.

## I. Reporting Templates and Failure Codes

A complete cellwise report should contain the following columns: cell name, target-mass estimate and interval, anchored-source mass estimate and interval, support flag, verifier-loss estimate, target-loss estimate if measured, discrepancy upper bound, formalization-gap upper bound, concentration term, and contribution to the full certificate. A global panel should report $p_0$, $\alpha_{\mathrm{unsup}}$, $\gamma_{\mathcal{C}}$, the pooled source verifier loss, the target-weighted supported-cell term, and the final upper bound.

Failure codes should be explicit:

| | |
|---|---|
| OUTSIDE-COVER | target item belongs to $C_0$, |
| UNSUPPORTED | $q_j = 0 < p_j$ in the population, |
| OPERATIONAL-UNSUPPORTED | finite-audit support rule fails, |
| LOW-COUNT | conditional count below the predeclared minimum, |
| HIGH-DISCREPANCY | $d_j^+$ dominates the cell contribution, |
| FORMALIZATION-UNAUDITED | $\epsilon_{\mathrm{form},j}$ has no quantitative audit, |
| QC-OR-NO-CALL | QC failure, censoring, assay infeasibility, or no-call; endpoint missing, |
| ADAPTIVE-LEAKAGE | final audit data influenced construction or selection. |

The conclusion of a benchmark report should state which codes remain. A low pass rate may reveal a weak prover; a high pass rate with unresolved failure codes reveals an incomplete target-risk claim.

### I.1. Cover sensitivity

The partition is part of the estimand, so a serious audit should report sensitivity to coarsening and refinement. A coarse partition may hide a missing submode inside a broad supported cell. An extremely fine partition may mark many cells unsupported even when a coarser target claim is reasonable. The report should therefore predeclare a primary partition and one or two sensitivity partitions. If the conclusion depends on a particular granularity, that dependence is itself a result.

Cover sensitivity should not be used post hoc to find the most flattering denominator. The correct practice is to state the target claim first and then choose the partition that reflects it. For example, a benchmark claiming competence on elementary algebra can use a different partition than a benchmark claiming broad mathematical reasoning. The same verifier pass rate may support the first claim and fail to support the second.

### I.2. Global Reporting Table

| Global quantity | Reported value or interval |
|---|---|
| Target population $P$ | Sampling frame, dates/version, exclusions. |
| Mode partition $\mathcal{C}$ | Cell definitions, annotators, overlap rule. |
| Residual mass $p_0$ | Estimate and simultaneous interval. |
| Population unsupported mass $\alpha_{\mathrm{unsup}}$ | Cells with $q_j = 0 < p_j$ and total target mass. |
| Pooled source verifier loss | Clearly labeled as source-weighted. |
| Target-weighted supported verifier term | Uses $p_j$, not $q_j$. |
| Discrepancy contribution | Per-cell upper bounds and class assumptions. |
| Formalization contribution | Rubric, sample size, adjudication, upper bounds. |
| Final certificate | Full-target upper bound or unresolved. |

This table is deliberately operational: a benchmark builder should be able to implement it. A theorem that cannot be turned into such a table is unlikely to change evaluation practice.

## J. Stress-Test Designs

The main text lists four stress tests. Here we specify their estimands more explicitly. In an uncovered-mass test, the intervention changes whether a target cell is present in the source. The primary estimand is the difference in target loss on that cell, evaluated on a target holdout not used to construct the source. Matching on source size and verifier loss is important because otherwise the test confounds support with overall training budget or prover quality.

In a mode-weight test, every named cell remains supported, but the source weights $q_j$ are shifted. The target-weighted estimand is

$$D = \sum_j (p_j - q_j)\widehat{R}_{T,j},$$

where $\widehat{R}_{T,j}$ is the held-out target-loss estimate in cell $C_j$. The report should also show the source-weighted version, because the two can move in opposite directions. That divergence is exactly why denominator labels matter.

In a within-mode transfer test, the support flag is held fixed while examples inside a cell become more target-like. The

natural estimands are the target-loss contrast inside $C_j$ and the change in the empirical discrepancy estimate. This test asks whether the cell was merely present or actually representative of the target conditional law.

In a curriculum allocation audit, the final theorem does not license a path claim by itself. The path $Q_0, \ldots, Q_T$ can be plotted as a diagnostic trajectory of support, weights, discrepancies, and formalization gaps. A claim that curriculum order has a causal effect on target risk requires a randomized or otherwise identified design and a stated sequential estimand.

## K. Reviewer Checklist for the Theorem Claim

A reader can evaluate the central claim with seven questions.

1. Is the title read as model-collapse support loss under ground-truth verification, not as "verification is useless"?
2. Are $\mathcal{X}$, $\mathcal{Z}$, $h_X$, $h_Z$, and $\tau$ typed separately?
3. Is the target partition predeclared, and are $p_j$ and $q_j$ raw masses?
4. Are unsupported cells charged by target mass instead of removed by normalization?
5. Is the discrepancy term tied to the target loss class?
6. Is formalization error audited separately from Lean soundness?
7. Is the final guarantee fixed-model, independent-holdout, and nonadaptive unless a correction is stated?

If the answer to all seven questions is yes, the theorem supports the bounded target-risk reading. If any answer is no, the theorem does not license that reading.

## L. Finite-Class Discrepancy Lemma

For completeness we record the finite-class version of the discrepancy estimate used by the ledger. It is deliberately elementary: the purpose is to make the certificate auditable by a benchmark builder rather than to optimize constants.

**Lemma 1.** *Let $\mathcal{F}_j = \{x \mapsto L_X^h(x) : h \in \mathcal{H}_0\}$ be finite and bounded in $[0,1]$. Draw independent samples $X_1, \ldots, X_n \sim P_j$ and $\widetilde{X}_1, \ldots, \widetilde{X}_m \sim S_j$. Let*

$$\widehat{d}_j = \sup_{f \in \mathcal{F}_j} \left| \frac{1}{n} \sum_{i=1}^{n} f(X_i) - \frac{1}{m} \sum_{i=1}^{m} f(\widetilde{X}_i) \right|.$$

*Define $d_{\mathcal{F}_j}(P_j, S_j) = \sup_{f \in \mathcal{F}_j} |\mathbb{E}_{P_j} f(X) - \mathbb{E}_{S_j} f(X)|$. Then, with probability at least $1 - \eta$,*

$$d_{\mathcal{F}_j}(P_j, S_j) \le \widehat{d}_j + \sqrt{\frac{\log(4|\mathcal{F}_j|/\eta)}{2n}} + \sqrt{\frac{\log(4|\mathcal{F}_j|/\eta)}{2m}}.$$

*Proof.* For a fixed $f$, Hoeffding's inequality controls the deviation of the target empirical mean from $\mathbb{E}_{P_j} f$, and separately controls the deviation of the anchored-source empirical mean from $\mathbb{E}_{S_j} f$. A union bound over both samples, both deviation directions, and all $f \in \mathcal{F}_j$ gives the two displayed radii simultaneously. The triangle inequality then gives the IPM bound. If the evaluated system or loss class was selected after seeing these samples, the finite class must include every selected candidate, or the audit must use a fresh split. □

This lemma is crude but useful. It shows exactly where the complexity term enters a verifier-supervised certificate: not in the Lean labels, which are already exact on source artifacts, but in the transfer of a loss class from $S_j$ to $P_j$. A sharper VC, Rademacher, stability, or kernel argument can replace this lemma when the report states the required assumptions.

## M. Support Decisions Under Small Source Mass

Support is a statistical decision as well as a population property. If $q_j$ is estimated from $N$ independent anchored-source samples and the observed count is $K_j$, then a cell should be marked supported only when the lower confidence bound for $q_j$ is positive and the protocol can draw the conditional examples required by the verifier-loss, discrepancy, and formalization audits. A point estimate $\widehat{q}_j > 0$ is not enough if the final conditional audit has no usable examples.

For small $q_j$, unconditional source auditing is inefficient. If $K_j \sim \mathrm{Binomial}(N, q_j)$, then

$$\Pr(K_j = 0) = (1 - q_j)^N.$$

This formula is the finite-sample version of near collapse. The source may have nonzero population support, but the audit can still miss the cell with high probability unless $Nq_j$ is large. In that case the benchmark may use stratified source collection from a documented frame, but it must report the distinction between the training source law, the audit sampling frame, and the weights used in the target-risk certificate.

The lower-bound message is unchanged. Perfect verifier labels remove uncertainty about the sampled source artifacts. They do not remove uncertainty about target-heavy cells that the artifact sampler rarely or never reaches. The ledger

therefore treats small-source-mass cells conservatively: either document a conditional audit design or charge the target mass as unsupported for the full-target claim.

## N. Proof Libraries, Autoformalization, and Adapter Scope

Formal libraries are not homogeneous sources. They have local conventions, imported theories, naming patterns, typeclass behavior, proof idioms, and dependency graphs. A source law $Q$ over library statements therefore induces a structured distribution in $\mathcal{Z}$. The adapter $\tau$ may map several syntactically different formal statements to one informal target mode, or it may fail when implicit arguments, typeclass instances, or imports affect meaning. These facts are not defects in the library; they are the reason an evaluation must expose its source frame.

For Lean-style environments, the kernel and library infrastructure give manifest-relative formal meaning to accepted artifacts. The ledger begins after that precision. It asks whether the sampled artifact is the right evidence for the target task. In a retrieval setting, $\tau$ may include retrieved premises and proof-state context. In autoformalization, it should record the informal statement and assumptions that the elaborated theorem statement is claimed to represent. In process-verified reinforcement learning, it may include proof-state trajectories. Each setting changes the formalization audit and discrepancy class, but none removes the need to report $p_j$, $q_j$, unsupported mass, and target-weighted uncertainty.

This is also why the theorem is not anti-verification. Verified labels identify acceptance of formal artifacts under the declared verifier environment and admission policy. When the declared target law is covered by the artifact law through an audited adapter, verification can be nearly ideal supervision. The negative result appears only when a target claim reaches beyond the semantic and support cover of the verified source. The right conclusion is not to weaken proof checking, but to pair it with explicit target denominators.

## O. Wet-Lab Assays as Protocol-Local Endpoint Verifiers

The same ledger logic applies when the verifier is an experiment rather than a proof assistant. A wet-lab assay can supply a protocol-local endpoint label. Examples include a sequencing pipeline calling a variant, a binding assay estimating affinity or supplying a thresholded affinity call, a cell-viability screen recording viability at a fixed dose and time point, or a preclinical study recording a prespecified phenotype under its protocol. For example, a binding assay may give control- and replicate-qualified endpoint calls under prespecified thresholds for assayed compounds in two cell lines. It still says little about target-positive strata never assayed: primary-cell donors, disease genotypes, exposure schedules, tissue contexts, or perturbations excluded by QC. QC-failed, censored, and no-call assay attempts are not endpoint negatives or positives; for a broad claim, they remain in the attempted-source denominator.

The claim becomes testable only after the denominator is named and the endpoint-to-target bridge is audited. Here $Q$ is the distribution of predeclared assay-attempt records, including attempts that fail QC, are censored, yield no call, or otherwise miss the endpoint; it is not only callable endpoint records and not the biological population itself. Each record should carry cell line, donor source, reagent lot, preparation, dose schedule, time point, instrument, threshold, replicate design, normalization rule, limit of detection or quantification, censoring or no-call rule, batch structure, and reason for any missing endpoint. The target law $P$ is the declared deployment or scientific population, with inclusion and exclusion rules fixed before endpoint outcomes are inspected. The adapter $\tau$ states the surrogate link; the audit must bound that bridge before the endpoint is treated as evidence about, not a replacement for, the target outcome. Examples include using an in vitro endpoint as an audited surrogate for an in vivo, clinical, ecological, or deployment-facing outcome. Unsupported mass corresponds to target subgroups, protocols, contexts, or perturbations that the experiment never reaches.

Thus "ground truth" should mean protocol-local truth: the endpoint label is ground truth for this assayed well or specimen under this protocol, not truth for the biological or deployment population. A QC-qualified wet-lab endpoint can make source-label error auditable for sampled wells or specimens, just as Lean removes source-label noise for sampled formal artifacts. It does not by itself prove that the sampled protocols cover the target population, that batch effects are irrelevant, that the assay endpoint is the right surrogate for the target loss, or that rare subgroups have been measured. QC exclusions, assay-infeasible samples, and no-call records must be reported in the source-attempt denominator and charged when the target claim includes them; they should not disappear from the target claim merely because no endpoint was observed. In ledger terms, the assay supplies $L_Z^h$; the report still owes $p_j, q_j$, discrepancy, endpoint-to-target audit, dependence-aware uncertainty, and a charge for unassayed target mass.

The analogy clarifies the scope. The issue is broader: a high-integrity measurement channel makes the remaining validity assumptions more visible. Whether the channel is a kernel check or a wet-lab assay, verified labels are strongest when the report says what source population they measure and where they stop.

# P. Additional Lower-Bound Details

This appendix records the precise decision-theoretic form of the unsupported-cell argument. Fix a target-positive cell $C_j$ with $p_j = P(C_j) > 0$ and $q_j = S(C_j) = 0$. Let $\mathcal{T}_n$ denote the transcript generated by any source-only procedure that observes $n$ source artifacts, their verifier outcomes, and any deterministic or randomized function of those outcomes, but does not observe target samples or query $h_X$ on target points. Since $Q(\tau(Z) \in C_j) = 0$, the law of $\mathcal{T}_n$ is unchanged by any modification of $h_X$ on $C_j$, as long as $h_Z$ and the target head outside $C_j$ are held fixed.

Choose two worlds $w = 0, 1$. In both worlds the artifact head, verifier loss, source sampler, adapter, and all observed source-side quantities are identical. The target head differs only on $C_j$: in world 0 the target loss is 0 on $C_j$, and in world 1 the target loss is 1 on $C_j$. Outside $C_j$, the target losses agree. Then

$$\mathsf{R}_P^{(1)}(h) - \mathsf{R}_P^{(0)}(h) = p_j.$$

Any estimator $\phi(\mathcal{T}_n)$ must output the same distribution in the two worlds. Therefore

$$\max_{w \in \{0,1\}} \mathbb{E}_w \left| \phi(\mathcal{T}_n) - \mathsf{R}_P^{(w)}(h) \right| \geq \frac{p_j}{2},$$

and for deterministic point estimates the same conclusion follows by the midpoint argument. If the theorem is stated as a difference between two systems rather than two worlds for one system, the construction simply assigns $h^0$ and $h^1$ to agree on the source support and disagree on $C_j$. The source transcript still cannot distinguish them.

The lower bound is distribution-free because it does not assume parametric structure tying $C_j$ to the observed source cells. A representation model, language prior, or informal semantic argument can be valuable scientific evidence, but it is an additional assumption. The theorem's role is to say what verified source labels identify without that assumption.

### P.1. Randomized procedures and adaptive source queries

Randomization does not change the unsupported-cell conclusion. Condition on the algorithm's internal randomness. For each realized random seed, the source transcript has the same law in the two worlds, so the deterministic argument applies. Averaging over seeds gives the displayed minimax inequality.

Adaptive source queries also do not help unless the query mechanism can leave the anchored source support. If each queried artifact is drawn from a rule whose support is contained in $\{z : \tau(z) \notin C_j\}$, then every adaptive transcript remains independent of the target behavior on $C_j$. To repair the failure, the benchmark must add a new measurement channel: target sampling, stratified source construction that reaches the cell, an audited semantic coverage assumption, or a formalization protocol that makes the cell reachable.

### P.2. Near-collapse as a local minimax event

When $0 < q_j \ll p_j$, the exact zero-support lower bound is replaced by a high-probability noncoverage event. If $N_j \sim \mathrm{Binomial}(N, q_j)$, the event $N_j = 0$ has probability $(1 - q_j)^N$. Conditional on this event, the audit transcript contains no source evidence from $C_j$, so the same indistinguishability argument applies for that realized audit. Thus any finite audit has a local ambiguity term whose probability is controlled by the actual source mass and the audit design. Stratification can reduce this term, but only by making the sampling frame and weights explicit.

This is why the main text separates population unsupported mass from finite-sample support scarcity. The former is an identification charge; the latter is an audit-design charge. Both are hidden by a pooled verifier pass rate, and both become visible in the ledger.

# Q. Arithmetic for a Full Certificate

This section gives a second schematic arithmetic example. Suppose a target partition has residual mass $p_0$, population unsupported mass $\alpha_{\mathrm{unsup}}$, and supported cells $J$. For each supported cell let

$$b_j = \widehat{r}_j + d_j^+ + \epsilon_{\mathrm{form},j}^+ + \rho_j.$$

The certificate reports

$$U(h) = p_0 + \alpha_{\mathrm{unsup}} + \sum_{j \in J} p_j b_j.$$

The source pass rate is not $1 - U(h)$. It is a source-weighted statistic,

$$1 - \sum_j q_j \widehat{r}_j,$$

and its weights are $q_j$, not $p_j$. The difference between these two weighted sums is the denominator problem in arithmetic form.

For example, take two supported cells and one unsupported cell:

| cell | $p_j$ | $q_j$ | $\widehat{r}_j$ | $d_j^+ + \epsilon_{\mathrm{form},j}^+ + \rho_j$ |
|---|---|---|---|---|
| 1 | .55 | .70 | .04 | .06 |
| 2 | .25 | .20 | .10 | .09 |
| 3 | .15 | 0 | – | – |
| $C_0$ | .05 | – | – | – |

Then $\alpha_{\mathrm{unsup}} = .15$, and

$$U(h) = .05 + .15 + .55(.10) + .25(.19) = .3025.$$

The pooled source verifier loss on the supported cells is $.70(.04) + .20(.10) = .048$, before accounting for the remaining source mass. Both numbers can be correct at the same time because they answer different questions. The first is a conservative full-target upper certificate under the stated audit terms. The second is a source-space verifier statistic.

The example also shows where improvement can occur. Better proof generation lowers $\widehat{r}_j$. Better source design or autoformalization can move target mass from $\alpha_{\text{unsup}}$ into supported cells. Better semantic matching lowers $d_j^+$ or $\epsilon^+_{\text{form},j}$. Larger independent audits lower $\rho_j$. These changes should not be collapsed into one leaderboard number, because they represent different scientific accomplishments.

## R. From Population Terms to a Reported Upper Bound

The theorem is written with population quantities so that the estimand is clear. A report, however, must give finite-audit upper bounds. The safest rule is to replace every unknown favorable quantity by a simultaneous conservative value before adding the terms. For target masses, use an upper confidence bound for charged cells and a lower confidence bound only when declaring support. For verifier losses, use the empirical conditional loss plus the holdout radius. For discrepancy and formalization terms, use upper confidence bounds produced by the prespecified audit. The final displayed certificate is then the maximum right-hand side over the joint confidence set, or an algebraically simpler upper bound that is valid on the same event.

This rule matters because different terms are estimated from different data sources. Target masses may come from a labeled target audit. Anchored source masses may come from a source-frame audit. Verifier losses come from conditional source holdouts. Discrepancy may require target labels or target-loss proxies. Formalization gaps come from semantic adjudication. Pooling these measurements into a single standard error would hide which uncertainty is responsible for the final number. The ledger instead keeps the uncertainty local until the final summation.

One conservative construction is the following. Allocate confidence budgets $\eta_p, \eta_q, \eta_r, \eta_d, \eta_f$ whose sum is at most $\eta$. Build simultaneous intervals for all $p_j$ under $\eta_p$ and all $q_j$ under $\eta_q$. A named cell is operationally supported only if its lower interval for $q_j$ is positive and the conditional audit has the required count. Otherwise its upper interval for $p_j$ is charged. For each supported cell, add the upper confidence bound for verifier loss, the upper discrepancy bound, the upper formalization-gap bound, and the displayed concentration radius. This is not the tightest possible certificate, but it is transparent: the ledger shows which event supports each term.

The same recipe handles partial audits. If a benchmark does not estimate a discrepancy term, it should not silently set it to zero. It can either narrow the claim to a setting where the discrepancy is assumed by design and state that assumption, or it can report the target-risk certificate as incomplete. If a formalization audit is absent, the formalization term is unaudited rather than small. This fail-closed behavior is the main difference between the ledger and a leaderboard score. A leaderboard number often rewards the best available aggregate. A target-risk certificate refuses a claim when a required measurement channel is missing.

## S. Formalization Gap Audit Details

The formalization term is easy to dismiss and risky to omit. A Lean proof can be perfectly valid for a theorem that is not the theorem the target task meant. The audit therefore compares three objects: the intended target task, the target object produced by the adapter $\tau(z)$, and the formal statement actually checked by the verifier. The adjudicator is not asked whether the proof is valid; the verifier already answered that. The adjudicator is asked whether the verified statement is the right semantic evidence for the target loss.

A minimal rubric has seven outcomes. CORRECT-FORMALIZATION means the formal statement preserves the target task under the stated assumptions. MISSING-ASSUMPTION means the formal statement proves a version with an unstated or stronger premise. WRONG-TARGET means the formalized theorem solves a different task. AMBIGUOUS-ADAPTER means the mapping from $z$ to $\tau(z)$ is not uniquely interpretable under the protocol. LABEL-MISMATCH means the target label $y(x)$ and the verifier objective disagree. DEPENDENCY-MISMATCH means imported facts or library conventions change the intended task boundary. UNADJUDICATED means the item could not be audited and must be charged conservatively.

To turn the rubric into $\epsilon^+_{\text{form},j}$, the report needs a loss-level mapping. The simplest mapping charges one unit for every outcome except CORRECT-FORMALIZATION. A refined mapping may charge only the maximum possible loss change implied by the category, but then the category-to-loss rule must be fixed before model losses are unblinded. Two independent annotators and an adjudicator are preferable because the formalization gap is a semantic measurement, not a deterministic compiler result. The audit should report disagreement rates as well as the final upper bound.

Formalization error is not a philosophical caveat. It is a measurable channel between the formal artifact and the target task. If it is small, the claim has stronger evidence. If it is large, the benchmark has learned something useful: the barrier is not proof search but semantic alignment. If it is

not measured, the target-risk claim remains incomplete no matter how high the verifier pass rate is.

## T. Nonadaptive Use of the Final Audit Split

The main theorem is stated for a fixed evaluated pipeline. This appendix spells out the common finite-selection extension and the reason it is not a license for open-ended benchmark redesign. Suppose a construction split is used to build adapters, prompts, rubrics, and mode definitions. A separate selection split chooses one pipeline from a finite set $\mathcal{H}_0$. The final audit split is then used once to certify the chosen pipeline. In the limited case where the only final-split adaptation is choosing one $h \in \mathcal{H}_0$, the verifier-loss radius in a supported cell becomes

$$\rho_j(\eta, \mathcal{H}_0) = \sqrt{\frac{\log(|J||\mathcal{H}_0|/\eta)}{2n_j}},$$

and the same union-bound proof gives simultaneous validity over all $h \in \mathcal{H}_0$ and all supported cells.

This correction is intentionally narrow. It does not cover changing the target distribution, inventing new cells, revising $\tau$, altering the formalization rubric, or choosing a discrepancy class after inspecting final audit failures. Those operations change the estimand or the loss class, not just the identity of $h$ in a finite list. They must be paid for by fresh data, sample splitting, reusable-holdout methods, or a separate uniform theorem.

**Lemma 2.** *Let $\mathcal{H}_0$ be a finite set fixed before the final audit split. For each $j \in J$ and $h \in \mathcal{H}_0$, let $\widehat{r}_j(h)$ be the mean of $n_j$ independent bounded verifier losses from $Q_j$. Then, with probability at least $1-\eta$, simultaneously for all $j \in J$ and $h \in \mathcal{H}_0$,*

$$\mathbb{E}_{Q_j} L_Z^h(Z) \leq \widehat{r}_j(h) + \sqrt{\frac{\log(|J||\mathcal{H}_0|/\eta)}{2n_j}}.$$

*Proof.* For fixed $j$ and $h$, Hoeffding's inequality gives a tail probability at most $\eta/(|J||\mathcal{H}_0|)$ for the displayed failure event. A union bound over cells and candidate pipelines proves the simultaneous claim. Substituting the simultaneous upper bounds into the supported-cell decomposition yields the finite-selection certificate. □

The final audit split can verify a finite menu that was already on the table. It cannot be used as a scratchpad for discovering the denominator. If the target partition, adapter, or formalization categories change after final audit inspection, the reported number is exploratory no matter how exact the verifier labels are.

## U. Broader Impacts and Responsible Use

The positive impact of the ledger is interpretability of benchmark claims. It helps benchmark builders, readers, and downstream users distinguish a verified artifact statistic from a target-risk claim. In domains where formal proofs are used to justify model reliability, that distinction can prevent overconfident deployment based on a clean but narrow denominator.

The main negative risk is misuse as a superficial checklist. A benchmark could fill in the ledger with weak proxies, unaudited adapters, or post-hoc partitions and still present the result as target-risk certified. The remedy is to make the assumptions inspectable: predeclare the partition, name the target population, expose unsupported mass, and label unaudited formalization gaps. The work releases no new model or scraped dataset; it contributes a reporting protocol and theorem for interpreting verified benchmark claims.