# Outcome Is Not Verification: Auditing Hidden-State Verifiers with Counterfactual Local Validity

**Anonymous authors**
**Paper under double-blind review**

## Abstract

Hidden states can support strong *outcome readout* without thereby supporting reliable *process verification*. We make that distinction operational with an audit centered on counterfactual local validity: whether the same step remains valid when the underlying world changes. We instantiate the audit in two structured reasoning domains—arithmetic DAG execution and grounded Horn-style logic—using paired worlds, stored gold/recoverable/doomed traces, and strict same-step flips that hold the current-step text fixed. To isolate the target variable rather than the architecture, we keep the hidden-state extractor family and scorer class fixed and compare only supervision targets: final trace outcome, prefix outcome, local validity, and a flip-rank variant. Across domains, hidden states provide meaningful trace-level outcome readout, but outcome-supervised step scoring does not behave like process verification. Under the same canonical step pools, LocalValidity improves the strongest audit axes, especially strict same-step FlipAcc and PVS. The gains are not uniform across all tests: repaired arithmetic scale remains mixed on first-invalid-step localization, particularly on held-out renderer C, while logic scale remains strongly favorable to LocalValidity even on the same harder transfer slice. The resulting picture is not that local-validity supervision "wins everything," but that counterfactual local validity exposes a real outcome/process gap while showing that process-verification evidence is metric-dependent and transfer-sensitive even in structured reasoning.

## 1 Introduction

Hidden-state verifiers are increasingly used as if they measured whether a model is reasoning correctly. That assumption is stronger than the evidence usually warrants. A hidden state can encode that a trace is likely to end correctly without encoding whether the *current step* is valid for the *current world*. In this paper we audit that distinction directly.

Our starting point is simple. If a verifier is doing process verification, then it should notice when the same step becomes invalid after the world changes. If it merely reads out eventual answer commitment, it can look strong on aggregate correctness while missing that local change. We therefore center the paper on counterfactual local validity: whether a candidate step is valid with respect to the gold pre-step state of a particular world.

The audit is deliberately narrow and controlled. We work in two executable domains—stateful arithmetic DAGs and grounded Horn-style logic—where local validity can be computed exactly, where the same step text can flip from valid to invalid, and where we can store matched gold, recoverable-splice, and doomed-splice traces. We keep the extractor family fixed (teacher-forced pooled hidden states from Qwen3-4B) and the scorer class fixed (linear scorers) and vary only the supervision target. The main comparison is therefore not architectural; it is whether the target variable is final outcome, prefix outcome, or local validity.

This setup yields three main findings. First, hidden states do support strong trace-level outcome readout. Second, outcome readout is not process verification: under identical step pools and scorer class, LocalValidity consistently improves the strongest audit axes, especially strict same-step FlipAcc and PVS. Third, that positive evidence does not transfer uniformly across all tests. Repaired arithmetic scale remains mixed

on first-invalid-step localization, especially on held-out renderer C, whereas logic scale remains strongly favorable to LocalValidity on the same harder slice. The resulting contribution is therefore sharper than a generic "better verifier" paper. counterfactual local validity is useful precisely because it reveals both the outcome/process gap and the fact that process-verification evidence is metric-dependent.

This paper sits between process-supervision work and faithfulness work. Process-supervision studies show that intermediate-step supervision can outperform outcome-only supervision for reasoning models (Lightman et al., 2023). Faithfulness work shows that explicit reasoning traces can be plausible yet unfaithful to the model's actual computation (Turpin et al., 2023; Lanham et al., 2023). Recent hidden-state work shows that internal states can encode intermediate-answer correctness (Zhang et al., 2025). Our contribution is to connect these threads with an audit object that targets process verification directly while holding extractor family and scorer class fixed.

**Contributions.** We make four contributions. (1) We formalize the distinction between *outcome readout* and *process verification* for hidden-state verifiers. (2) We introduce counterfactual local validity and strict same-step flips as step-level audit objects that control current-step text while changing the world. (3) We provide paired arithmetic and grounded-Horn audit settings with gold, recoverable, doomed, and held-out-renderer slices. (4) Empirically, we show that hidden states support strong outcome readout, but the strongest evidence for process verification comes from strict same-step FlipAcc and PVS; localization is a harder transfer test whose behavior is domain-sensitive rather than guaranteed.

## 2 Audit object and definitions

A *schema* fixes branch structure, slot identities, semantics, and allowed interleavings. A *world* instantiates that schema with concrete leaf values (arithmetic) or grounded facts (logic). One rendered line corresponds to one schema slot; this is the paper's notion of a *step*. For a world $w$ and slot $t$, let $s_t^*(w)$ denote the gold executor state after slot $t$.

Given a candidate structured step record $r_t$ placed at slot $t$ in world $w$, its local-validity label is

$$y_t(r_t, w) = \mathbf{1}\{r_t \text{ is correct relative to } s_{t-1}^*(w) \text{ and the slot semantics}\}. \tag{1}$$

The label is computed against the gold pre-step state of the target world, never against a trace's own erroneous state.

A *strict same-step flip* is the paper's headline audit object. It is a pair of step instances that share the same schema slot, renderer, and rendered text, but have opposite local-validity labels under different worlds. Formally, if $s(h_t)$ is a step score where larger means "more locally valid," then strict same-step flip accuracy is

$$\text{FlipAcc} = \frac{1}{N}\sum_{i=1}^{N} \mathbf{1}\big[s(h_i^{\text{valid}}) > s(h_i^{\text{invalid}})\big]. \tag{2}$$

Because the text is identical within each pair, this metric targets context-conditioned validity rather than lexical cues.

We use two additional headline metrics. Let $o \in \{0, 1\}$ be final-trace correctness. The process-verification score

$$\text{PVS} = \tfrac{1}{2}\Big(\text{AUROC}(s_t, y_t \mid o = 1) + \text{AUROC}(s_t, y_t \mid o = 0)\Big) \tag{3}$$

measures whether a scorer separates valid from invalid steps even after conditioning on final outcome. For first-invalid-step localization, each trace has

$$f(\tau, w) = \min\{t : y_t(r_t, w) = 0\}, \tag{4}$$

with $f = T + 1$ if no step is invalid. A scorer predicts the first invalid step by thresholding its step scores with a validation-chosen threshold and taking the earliest crossing.

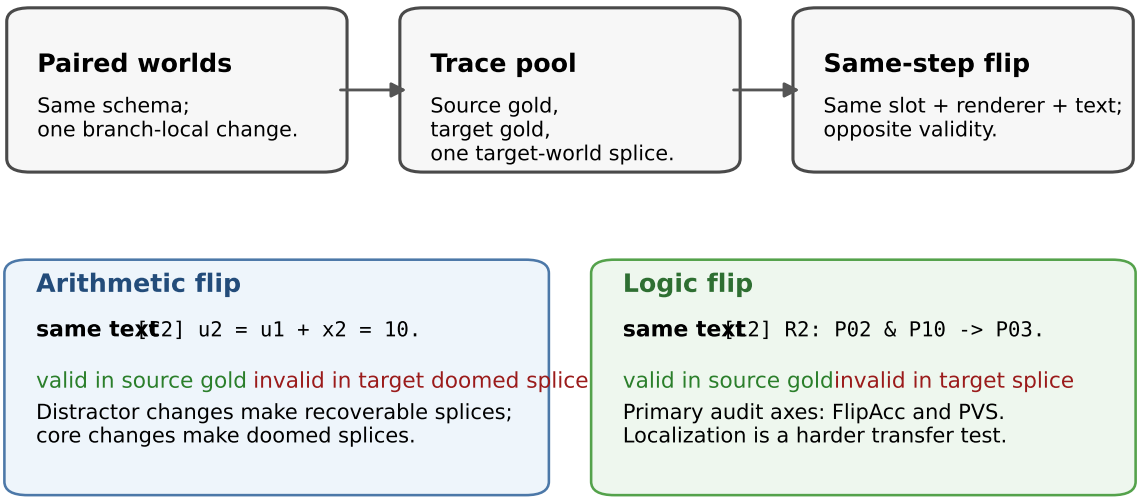


Figure 1: **Counterfactual local validity audit.** Paired worlds share a schema but differ by one branch-local input or fact family. For each world pair we store two gold traces and one target-world splice trace. Strict same-step flips reuse the same slot, renderer, and rendered text while changing the world-specific validity label. The examples illustrate the paper's primary control: the current-step text is fixed, but validity flips.

Finally, REB is a secondary diagnostic rather than a headline claim:

$$\mathrm{REB} = \mathbb{E}[s_t \mid y_t = 0, o = 1] - \mathbb{E}[s_t \mid y_t = 0, o = 0]. \tag{5}$$

Positive values indicate that invalid steps in correct traces still receive higher scores than invalid steps in incorrect traces, suggesting residual outcome dependence.

## 3 Counterfactual audit construction

**Domains.** Arithmetic uses 7 derived slots: three core steps $C1$–$C3$, three distractor steps $D1$–$D3$, and a final answer step `Ans` that depends only on the core branch. Logic uses 6 derived slots: three core rule applications $L1$–$L3$, two distractor applications $N1$–$N2$, and a final grounded answer step `Ans`. In both domains the prompt contains the leaf inputs or grounded facts; only derived steps are scored.

**World pairs and trace pool.** Each world pair shares a schema and changes exactly one branch-local input or fact family. For every pair we store three traces: `source_gold`, `target_gold`, and one target-world splice. If the changed branch is distractor, the splice is *recoverable*: distractor steps become invalid but the answer remains correct. If the changed branch is core, the splice is *doomed*: core steps (and the answer step) become invalid and the final answer becomes wrong. This construction guarantees nonempty invalid-recoverable and invalid-doomed sets in every split.

**Renderers and held-out transfer.** Each structured step record is rendered by one of three frozen renderers. Renderers A and B appear in training; at scale, renderer C is excluded from both training and validation and appears only in test, making the renderer-C slice claim-eligible held-out evidence. The current-step text changes with the renderer, but the underlying step record and validity label do not.

Table 1: Dataset summary. Counts are totals across both change branches before scorer-specific pooling, in train/val/test order. Mini-pilot numbers are reported over three seeds; scale uses one seed. Renderer codes use AB for train on renderers A+B and ABC for pooled test slices containing all three renderers.

| Domain | Stage | Seeds | World pairs | Step instances | Strict flips | Renderers |
|---|---|---|---|---|---|---|
| Arithmetic | Mini-pilot | 3 | 200 / 50 / 50 | 8.4k / 3.15k / 3.15k | 1.4k / 525 / 525 | AB/ABC/ABC |
| Arithmetic | Scale | 1 | 4800 / 480 / 480 | 201.6k / 20.2k / 30.2k | 33.6k / 3.36k / 5.04k | AB/AB/ABC |
| Logic | Mini-pilot | 3 | 200 / 50 / 50 | 7.2k / 2.7k / 2.7k | 1.2k / 450 / 450 | AB/ABC/ABC |
| Logic | Scale | 1 | 2400 / 480 / 480 | 86.4k / 17.3k / 25.9k | 14.4k / 2.88k / 4.32k | AB/AB/ABC |

**Mandatory integrity checks.** The audit is only meaningful if text and pool confounds are controlled. We therefore enforce: schema splits, exact-string balance in strict flips, text-only leakage checks, position-leakage checks, interleaving-balance checks, nonempty conditioning sets for PVS and REB, per-renderer flip support, and dataset-version-scoped pool and embedding provenance. The paper reports only repaired arithmetic datasets and provenance-hardened runs.

## 4 Hidden-state verifiers under audit

We extract hidden states from `Qwen3-4B-Instruct-2507` (Yang et al., 2025) under teacher forcing. For each step we take the last four decoder layers, average them, and mean-pool over the current step's token span. This produces one pooled hidden-state vector per stored step. The feature extractor family is fixed throughout the paper.

We compare four linear scorers.

- TraceOutcome: one final-step example per stored trace, supervised by final trace correctness.
- PrefixOutcome: every step in the canonical step pool, supervised by final trace correctness.
- LocalValidity: the same canonical step pool, supervised by local-validity labels.
- LocalValidity+FlipRank: the same canonical step pool with local-validity supervision plus a strict-flip ranking term.

The fairness invariant is explicit: PrefixOutcome, LocalValidity, and LocalValidity+FlipRank use the *same* step-instance universe; only their labels and losses differ. TraceOutcome uses the same stored trace pool collapsed to one final-step example per trace. The paper is therefore about the audit target variable, not about model capacity or optimizer differences.

## 5 Experimental setup

The arithmetic mini-pilot uses 200/50/50 world pairs (train/val/test) per seed after renderer expansion; arithmetic scale uses 4800/480/480 world pairs. Logic mini-pilot uses 200/50/50 world pairs; logic scale uses 2400/480/480. Table 1 summarizes world-pair, step-instance, and strict-flip counts after renderer expansion.

The primary evidence structure is as follows. Mini-pilot results are means and standard deviations across three seeds; scale results are seed 0 with the repaired, provenance-hardened pipeline. Our headline comparisons focus on PrefixOutcome vs. LocalValidity. LocalValidity+FlipRank and REB are reported, but intentionally treated as secondary. Held-out renderer C is a main stress slice for scale only, because only scale excludes renderer C from both training and validation.

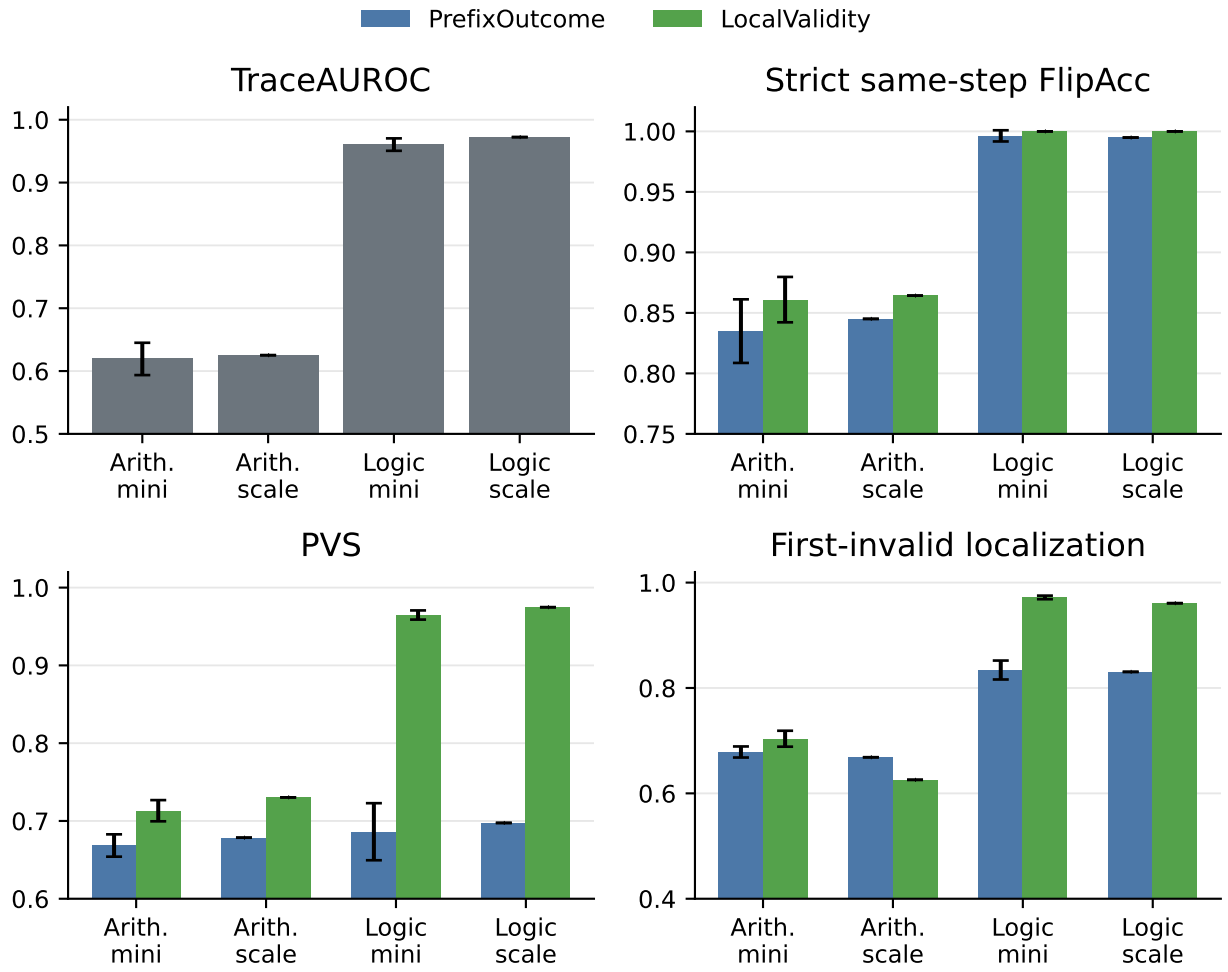


Figure 2: **Core pooled results.** Hidden states support outcome readout in both domains (TraceAUROC), but the stronger process-verification evidence comes from strict same-step FlipAcc and PVS. Local-validity supervision improves those metrics in arithmetic and logic. First-invalid-step localization is mixed: it improves in both mini-pilots and in logic scale, but reverses in repaired arithmetic scale. Error bars show standard deviation across three mini-pilot seeds.

## 6 Results

### 6.1 Hidden states support outcome readout

Trace-level outcome readout is clearly present in both domains. At scale, TraceOutcome reaches AUROC 0.625 in arithmetic and 0.972 in logic (Table 2; Figure 2). Logic is easier than arithmetic for this readout, but the qualitative point is the same: hidden states carry meaningful information about eventual trace correctness. That fact alone, however, does not establish process verification.

### 6.2 Outcome readout is not process verification

The strongest evidence for the paper's central distinction comes from strict same-step FlipAcc and PVS. Under the same canonical step pools and the same linear scorer class, LocalValidity beats PrefixOutcome on both metrics in every pooled setting we study (Figure 2; Table 2). The gap is modest but stable in arithmetic and much larger in logic. In the three-seed mini-pilots, LocalValidity also beats PrefixOutcome on both strict same-step FlipAcc and PVS in every arithmetic seed and in the aggregate logic mini-pilot summary

Table 2: Main pooled scale results. Boldface marks the better value between PrefixOutcome and LocalValidity within each row. Mini-pilot means and standard deviations appear in Figure 2 and Appendix C.

| Domain | TraceAUROC | Pref. Flip | Local Flip | Pref. PVS | Local PVS | Pref. Loc. | Local Loc. |
|---|---|---|---|---|---|---|---|
| Arithmetic | 0.625 | 0.845 | **0.864** | 0.679 | **0.730** | **0.669** | 0.626 |
| Logic | 0.972 | 0.995 | **1.000** | 0.697 | **0.975** | 0.831 | **0.961** |

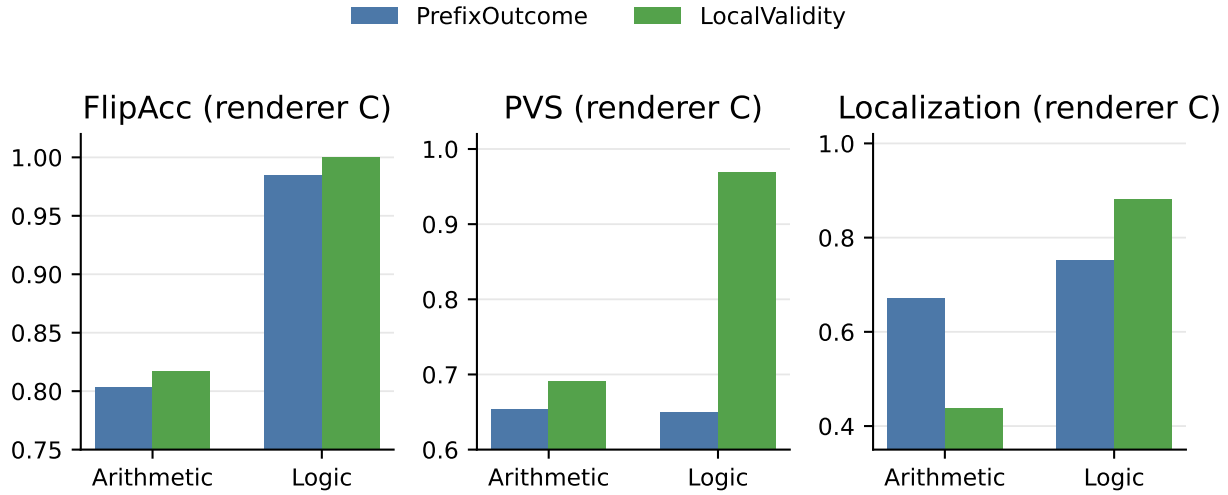


Figure 3: **Held-out renderer-C slice at scale.** Renderer C is excluded from both training and validation at scale. The renderer-C results preserve the main dissociation on strict same-step FlipAcc and PVS, but localization behaves differently across domains: arithmetic reverses, while logic remains favorable to LocalValidity. This is the clearest evidence that localization is a harder transfer test than the paper's primary audit axes.

(Appendix C). At scale, arithmetic moves from 0.845 to 0.864 on strict same-step FlipAcc and from 0.679 to 0.730 on PVS, while logic moves from 0.995 to 1.000 and from 0.697 to 0.975.

These gains are not artifacts of a richer model or a different feature extractor. The extractor family, pooling rule, scorer class, and step-instance universe are fixed; only the supervision target changes. The result is therefore the paper's core empirical point: strong outcome readout does not imply strong process verification.

### 6.3 Localization is a harder transfer test

The paper should *not* be framed as "LocalValidity wins everything." First-invalid-step localization is a materially harder test, and repaired arithmetic scale shows why. In pooled arithmetic scale, exact localization favors PrefixOutcome over LocalValidity (0.669 vs. 0.626), even though LocalValidity still wins on strict same-step FlipAcc and PVS. The held-out renderer C slice makes the reversal sharper: 0.671 for PrefixOutcome versus 0.438 for LocalValidity. Logic scale does not show the same failure mode. On the same held-out slice, logic remains strongly favorable to LocalValidity: strict same-step FlipAcc 1.000 vs. 0.985, PVS 0.969 vs. 0.650, and localization 0.883 vs. 0.753.

The correct interpretation is therefore metric-dependent rather than uniformly positive. counterfactual local validity exposes a real outcome/process gap, but stronger conditional local-validity discrimination does not automatically transfer to robust first-error localization under every renderer and domain.

### 6.4 FlipRank and REB are informative but secondary

LocalValidity+FlipRank often improves strict same-step FlipAcc, especially in arithmetic scale, but it does not provide a clean strengthening story. At scale, LocalValidity+FlipRank reaches the best arithmetic strict same-step FlipAcc (0.907 pooled, 0.873 on held-out renderer C), but it underperforms LocalValidity on PVS and localization. In logic scale, LocalValidity+FlipRank again trails LocalValidity substantially

on PVS and localization. REB is useful for diagnosing outcome dependence in arithmetic, but it is not a stable cross-domain headline result: in logic, both PrefixOutcome and LocalValidity have negative REB, and the sign does not admit the same simple reading. We therefore treat both diagnostics as secondary. Numerical details appear in Table 4 in the appendix.

**Seed stability.** Three-seed mini-pilots support the same qualitative picture. Arithmetic shows a stable LocalValidity > PrefixOutcome gap on strict same-step FlipAcc and PVS across all three seeds; logic preserves that pattern across the three-seed mini-pilot as well, with much larger PVS gaps. We include the seed-stability summary and gap plot in the appendix because they support the main story without changing its claim posture.

## 7 Discussion

The paper's main conclusion is not that local-validity supervision solves hidden-state verification. The stronger and more useful conclusion is methodological: counterfactual local validity distinguishes outcome readout from process verification, and once that distinction is made explicit, the strength of the evidence becomes metric-dependent. strict same-step FlipAcc and PVS are the most reliable axes in our setting because they directly measure whether a scorer tracks local validity rather than only eventual outcome. Localization is a harder transfer test. It can agree with the core story, as it does in logic, but it can also diverge under a held-out renderer and larger arithmetic scale.

That mixed result is not an embarrassment to be averaged away; it is the audit's main lesson. If a hidden-state verifier improves on one process-verification metric but fails on a stricter transfer-oriented metric, the right conclusion is not that the audit was too hard. The right conclusion is that process verification is not a single number. A verifier can be better at separating valid from invalid steps conditionally on outcome without yet being a robust localizer of the first invalid step under all renderer shifts.

The paper is intentionally scoped. We work only in structured, executable reasoning where local validity can be computed exactly. Teacher forcing is appropriate for scoring provided prefixes, but it does not justify claims about online monitoring during free generation. We also do not claim that counterfactual local validity exhausts reasoning faithfulness; it is one audit object, chosen because it makes the outcome/process distinction observable and testable.

## 8 Conclusion

Hidden states can support strong outcome readout without thereby supporting reliable process verification. Counterfactual local validity makes that distinction visible. In structured arithmetic and grounded Horn logic, local-validity supervision strengthens the clearest audit axes—strict same-step flips and PVS—under the same extractor family, scorer class, and step pools. At the same time, repaired arithmetic scale shows that these gains do not automatically transfer to first-invalid-step localization under held-out renderer shift, while logic scale shows that the harder test can still succeed in a second domain. Counterfactual local validity is therefore useful not because it yields a uniformly better verifier, but because it exposes where outcome readout ends and process verification begins.

## References

Tamera Lanham et al. Measuring faithfulness in chain-of-thought reasoning. *arXiv preprint arXiv:2307.13702*, 2023.

Hunter Lightman et al. Let's verify step by step. *arXiv preprint arXiv:2305.20050*, 2023.

Miles Turpin, Julian Michael, Ethan Perez, and Samuel R. Bowman. Language models don't always say what they think: Unfaithful explanations in chain-of-thought prompting. In *Advances in Neural Information Processing Systems*, 2023.

An Yang et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Anqi Zhang, Yulin Chen, Jane Pan, Chen Zhao, Aurojit Panda, Jinyang Li, and He He. Reasoning models know when they're right: Probing hidden states for self-verification. *arXiv preprint arXiv:2504.05419*, 2025.

## A Renderer-C numerical results

Table 3: Held-out renderer-C slice at scale. Boldface again compares PrefixOutcome and LocalValidity within each row. Renderer C is excluded from both training and validation at scale, so these are genuine held-out-renderer metrics.

| Domain | TraceAUROC | Pref. Flip | Local Flip | Pref. PVS | Local PVS | Pref. Loc. | Local Loc. |
|---|---|---|---|---|---|---|---|
| Arithmetic | 0.612 | 0.804 | **0.817** | 0.655 | **0.691** | **0.671** | 0.438 |
| Logic | 0.968 | 0.985 | **1.000** | 0.650 | **0.969** | 0.753 | **0.883** |

## B Secondary diagnostics

Table 4: Secondary diagnostics at scale. LocalValidity+FlipRank sometimes improves flip accuracy but does not preserve a clean PVS/localization advantage. REB is directionally informative in arithmetic but mixed in logic.

| Domain | Slice | +Flip Flip | +Flip PVS | +Flip Loc. | Pref. REB | Local REB | +Flip REB |
|---|---|---|---|---|---|---|---|
| Arithmetic | Pooled | 0.907 | 0.703 | 0.610 | 0.040 | 0.024 | 0.033 |
| Arithmetic | Renderer C | 0.873 | 0.693 | 0.455 | 0.050 | 0.039 | 0.027 |
| Logic | Pooled | 0.997 | 0.796 | 0.667 | -0.034 | -0.323 | 0.000 |
| Logic | Renderer C | 0.999 | 0.781 | 0.667 | -0.051 | -0.172 | 0.000 |

## C Seed stability summary

Table 5: Mini-pilot seed-stability summary. The core LocalValidity > PrefixOutcome story is stable on strict same-step FlipAcc and PVS; the table records whether the qualitative conditions hold per seed.

| Domain | Seeds | Core story present | Local > Prefix on FlipAcc | Local > Prefix on PVS |
|---|---|---|---|---|
| Arithmetic | 3 | 3/3 | 3/3 | 3/3 |
| Logic | 3 | 2/3 | 2/3 | 3/3 |

## D Implementation and reporting integrity

All reported main results use the repaired arithmetic datasets, deterministic seeded linear training, dataset-version-scoped embedding storage, canonical pool manifests, and explicit renderer-C extraction acceptance. Historical arithmetic artifacts generated before those repairs are retained for audit trail only and are not used as paper evidence.

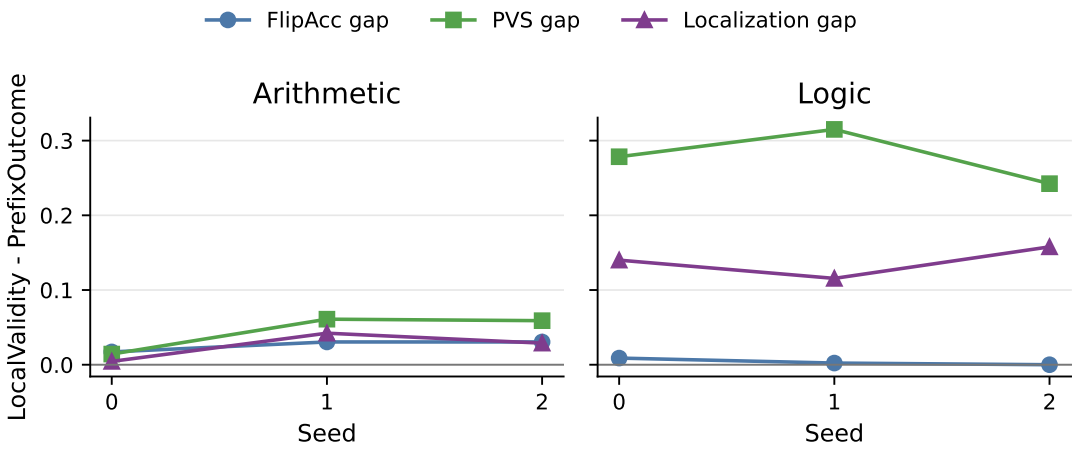


Figure 4: Mini-pilot metric gaps (LocalValidity minus PrefixOutcome) across seeds. The most stable backbone is the positive gap on strict same-step FlipAcc and PVS. Localization is positive in all mini-pilot seeds but becomes the harder transfer test at arithmetic scale.