DISSOCIATING INTERNAL ERROR DETECTION FROM STATED VERDICTS IN SMALL LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Small instruction-tuned models asked for a one-word verdict on whether an arithmetic derivation is correct give nearly the same answer whether or not it contains an error. We ask where the evidence of the error is lost. In 1,280 seeded derivations, half contain one wrong step and every later step is recomputed from the wrong value. In Qwen2.5-1.5B-Instruct, the model's own next-token surprisal separates corrupted from clean derivations with AUC 0.893 and finds the wrong step 79.2% of the time, while the Yes/No verdict on the same inputs reaches AUC 0.544. A consistent twin that writes the same numbers, with one operand changed so that every line is correct, is no more surprising than the clean derivation. In the same model, a linear probe on the final normalized state at the answer position reaches AUC 0.803, and above 0.76 at every distance between the error and the question, so the evidence is present in the state the verdict is read from and carries little weight on Yes against No. The probe scores the twins like clean derivations. Four worked examples in the prompt leave the verdict at 0.538, and asking the model to recompute every step in writing first raises the verdict AUC from 0.530 to 0.784 on a subsample of 256 derivations. Across seven models from three families (Qwen2.5 from 0.5B to 7B, Llama-3.2 1B and 3B, Gemma-2 2B), surprisal reaches AUC 0.859 to 0.931 and the verdict 0.503 to 0.605, whether a model accepts or rejects most derivations, and the final-state probe reaches 0.773 to 0.871 in Qwen2.5-1.5B and 7B, Llama-3.2-3B, and Gemma-2-2B and stays low in the two smallest models and in Qwen2.5-3B. On ProcessBench, the digit surprisal of Qwen2.5-Math-1.5B-Instruct, with no training and with one statistic and one threshold chosen on GSM8K, localizes the first error with an average F1 of 42.9. That is at or above 5 of the 6 process reward models evaluated in the benchmark paper and below the best of them (56.5) and a newer 7B model (73.5).

1 INTRODUCTION

Process supervision (Lightman et al., 2024), self-correction (Huang et al., 2024; Kumar et al., 2025), and verifier-guided search (Cobbe et al., 2021; Zhang et al., 2025b) all lean on the idea that checking a solution is easier than producing one (Cobbe et al., 2021; Song et al., 2025c). For the small models we study, it does not hold. Asked for a one-word verdict on a derivation, Qwen2.5-0.5B-Instruct and Qwen2.5-1.5B-Instruct accept every correct derivation and every corrupted one. Larger models also struggle to locate reasoning errors when asked (Tyen et al., 2024) and detect errors in model responses with low recall (Kamoi et al., 2024).

A model that fails to verify might not compute the relevant quantity, or might compute it and fail to use it. The two explanations call for different fixes, and output-level evaluation cannot tell them apart. We separate them with a controlled harness: seeded arithmetic and modular-arithmetic derivations of two to nine steps, each paired with a copy in which exactly one step's result is wrong and every later step is recomputed from the wrong value, so that only one line is locally inconsistent. Because the error position, the error size, and the derivation length are set by construction, every internal measurement can be read as a function of where the error is and how far it sits from the question.

We measure verification in three stages on the same problems. *Notice*: when the model reads the wrong result, does it assign it low probability? *Carry*: at later steps, does the residual stream still

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

encode that an error has occurred? *Report*: at the answer position, is the evidence present, and does the Yes/No verdict use it?

Our findings:

- Both models notice. The surprisal of the written result at the planted step is 8.1 to 8.2 nats higher than at the same step of the clean copy, and taking the most surprising step localizes the error 75% to 79% of the time. Asked which step is wrong, the same models localize 23.4% and 23.6% of errors, against 22.9% for a uniform guess. A consistent twin that ends the planted line in the same tokens costs no more than the clean copy there, and 7.3 to 7.5 nats less than the corrupted copy.
- Carrying is where the models differ. A probe for "an error has occurred so far" reads the 0.5B model's last layer at AUC 0.764 on the wrong step itself and near chance one step later. The 1.5B model keeps the evidence, decaying from 0.905 to 0.726 over five steps, and the 7B model keeps more (0.932 on the wrong step, 0.822 three steps later).
- At the answer position of the 1.5B model the evidence is present and barely used. A linear probe on the final normalized state reaches AUC 0.803, and 0.766 to 0.886 at each distance from the error, while the Yes/No log-odds computed from the same vector reach 0.544. The signal becomes linearly decodable within the last decoder block (a probe reads 0.572 before it and 0.802 after it), and the No-minus-Yes readout gives it little weight. The probe scores consistent twins like clean derivations, and four worked examples in the prompt leave the verdict at 0.538 while the same probe rises to 0.851. Asked to recompute every step before answering, the same model rejects about half of the corrupted derivations (verdict AUC 0.784 on a subsample where the one-word verdict reaches 0.530).
- The dissociation persists at every size and in every family we tested: Qwen2.5 from 0.5B to 7B, Llama-3.2 1B and 3B, and Gemma-2 2B. Surprisal separates corrupted from clean derivations at AUC 0.859 to 0.931 in all seven models, and the verdict stays between 0.503 and 0.605, whether a model accepts most derivations, as Qwen2.5 and Gemma-2 do, or rejects most of them, as Llama-3.2 does. The final-state probe reads 0.773 to 0.871 in Qwen2.5-1.5B and 7B, Llama-3.2-3B, and Gemma-2-2B, and stays low in the two smallest models and in Qwen2.5-3B, in fp16 and in fp32.
- The surprisal signal also localizes natural errors. On ProcessBench, Qwen2.5-Math-1.5B-Instruct's digit surprisal reaches an average F1 of 42.9 with no training, at or above 5 of the 6 process reward models evaluated in the benchmark paper and below the best of them. The same model's one-token answer to the same question scores 0.7.

Earlier work shows that probability measurements often beat prompted judgments, most of all in small models (Hu & Levy, 2023; Hu & Frank, 2024), that models often fail to validate answers they generate (Li et al., 2024; Song et al., 2025c), and that internal signals flag reasoning mistakes (Ye et al., 2025; Zhang et al., 2025a; Zhao et al., 2026). What we add is a harness that plants exactly one locally inconsistent line at a set position and size and measures noticing, carrying, and reporting on the same forward pass. With it we find the evidence linearly present in the state the verdict is read from in four of the seven models, from three families, with little weight on Yes against No, and we show that a training-free surprisal localizer from a 1.5B model reaches the level of early process reward models on ProcessBench.

2 SETUP

Problems. An arithmetic problem starts from an integer in $[2, 50]$ and applies additions, subtractions, and multiplications (multipliers 2 to 9, addends 2 to 40, multiplication suppressed above 400). A modular problem fixes a prime modulus in $\{7, 11, 13\}$ and applies additions and multiplications, reducing after each step. Depth runs from 2 to 9 steps; we draw 40 problems per family and depth from a fixed seed, 640 problems in total. Each problem is rendered as a derivation with one line per step ("Step 3: 17 + 8 = 25") and a final answer. The corrupted copy replaces the result of one uniformly chosen step with a wrong value, off by 1 to 3 (subtle, half the problems) or by 10 to 60 (gross), and recomputes every later step from the wrong value. Each problem therefore yields two derivations that differ from the planted step onward and are internally consistent everywhere except that step.

Models and prompt. We use Qwen2.5-0.5B-Instruct and Qwen2.5-1.5B-Instruct (Qwen et al., 2024) in fp32 with the chat template. The prompt contains the problem, the derivation, and the question "Is every step of this derivation arithmetically correct? Answer with one word: Yes or No." The verdict score is the log-probability of the Yes tokens minus the No tokens at the answer position. The harness measurements take one forward pass per derivation (1,280 per model) on a single T4 GPU.

Stage measurements. We use *notice*, *carry*, and *report* as names for three measurements made on the same forward pass; each is defined by its readout and makes no claim about what the model represents or intends. *Notice* is the summed surprisal of the tokens of each step's written result under teacher forcing. We score a derivation by its most surprising step relative to the derivation's median, and localize the error at the most surprising step. *Carry* is a logistic probe on the residual stream at the last token of each step's result, trained to predict whether a wrong step has occurred at or before that step. We evaluate it at each offset t after the error against clean-derivation positions with the same step index. *Report* has two readouts at the answer position: the model's own Yes/No log-odds, and a logistic probe on the hidden state. The verdict rejects a derivation when its Yes-minus-No margin is at most zero, which we call the default threshold. Probes are ℓ_2-regularized logistic regressions with standardized features, evaluated out of fold with five folds grouped by problem, so the clean and corrupted copies of a problem never fall on opposite sides of a split. Sections 6 and 7 repeat the stage measurements across model size and family, and Section 8 repeats the notice measurement and a verbal readout on natural errors.

3 SURPRISAL MARKS THE WRONG STEP

At the planted step, the written wrong result costs Qwen2.5-0.5B 8.10 more nats than the correct result at the same step of the clean copy (95% bootstrap CI 7.53 to 8.69) and costs Qwen2.5-1.5B 8.21 more nats (7.65 to 8.73). The corrupted copy is the more surprising one in 83.8% and 85.8% of pairs, and the surprisal at the error step alone separates corrupted from clean derivations with AUC 0.934 and 0.948. Gross errors cost more than subtle ones (9.32 versus 6.88 nats for 0.5B, 9.05 versus 7.38 for 1.5B), but subtle errors are still expensive.

Without knowing where the error is, scoring each derivation by its most surprising step gives AUC 0.872 (0.5B) and 0.893 (1.5B), and the most surprising step is the planted step in 75.2% and 79.2% of corrupted derivations. Localization is best when the error is in the first quarter (0.851 and 0.887) and roughly flat after that (0.69 to 0.76). Asked which step is wrong, with the answer read as the most probable step-number token, the same models localize 23.4% and 23.6% of errors, against 22.9% for a uniform guess, and guess late steps: their mean predicted relative position is 0.807 and 0.738 against a true mean of 0.495 (Appendix A).

After the planted step, surprisal returns to the clean-copy level within one step (Appendix C), so the models condition on the written wrong value and continue from it.

Consistent twins. A wrong result is also an unusual number, so part of the gap could come from the tokens themselves. For 468 of the 640 problems we build a consistent twin of the corrupted copy. It writes the same values from the planted step on and changes one operand at that step so that the line is correct ("Step 1: 8 + 60 = 68" against the corrupted "Step 1: 8 + 23 = 68"; the problem statement changes to match), so at the planted step the twin and the corrupted copy end in identical tokens. For Qwen2.5-0.5B and 1.5B those tokens cost 7.28 and 7.50 more nats in the corrupted copy than in the twin (95% CIs 6.65 to 7.95 and 6.94 to 8.08), and in both models the twin's result costs 0.04 nats less than the correct result of the clean copy. Scored by its most surprising step, the twin is indistinguishable from the clean derivation (AUC 0.504 and 0.513). Surprisal rises when a result disagrees with its operands, whatever the number. In the three models from other families (Section 7) and in Qwen2.5-3B, the corrupted copy costs 7.82 to 9.07 more nats than its twin, and the twin again scores like the clean derivation (AUC 0.510 to 0.519).

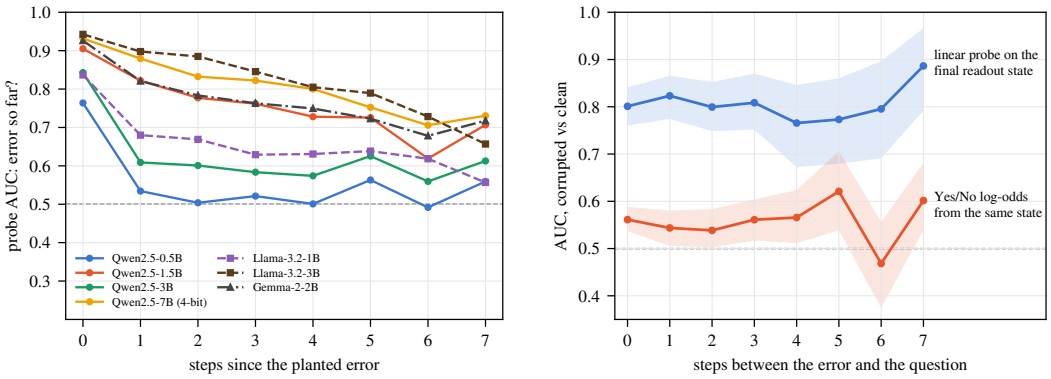

Figure 1: Left: last-layer probe for "an error has occurred so far" as a function of steps since the error, for every model; weight precision is given where it is not fp32. Right: Qwen2.5-1.5B at the answer position, by the number of steps between the error and the question, with 95% bootstrap intervals over problems. The probe on the final normalized state stays between 0.77 and 0.89 at every distance; the Yes/No log-odds computed from the same state stay between 0.47 and 0.62.

4 Carrying the evidence is where the models differ

Figure 1 (left) shows the carry probe on the last layer. In Qwen2.5-0.5B it detects the error at the wrong step itself (AUC 0.764) and falls to 0.534 one step later and to chance after that. In Qwen2.5-1.5B it starts at 0.905 and decays slowly: 0.822, 0.777, 0.762, 0.728, and 0.726 over the next five steps. Qwen2.5-7B keeps more: 0.932 on the wrong step, 0.822 three steps later, and 0.731 seven steps later. Qwen2.5-3B behaves more like the 0.5B model after the first step: 0.843 on the wrong step, then 0.609 and 0.584 one and three steps later, in fp16 and, within 0.001, in an fp32 rerun (Section 6). Among the other families, Llama-3.2-3B keeps the most three steps after the error (0.845) and Llama-3.2-1B the least (0.629; Section 7). Probes on earlier layers carry less, so the evidence that survives lives mostly in the final layer (Appendix C).

5 At the answer, the evidence is present and weakly read out

Presence. At the answer position, a probe on Qwen2.5-0.5B's hidden state reaches AUC 0.554 at best over 15 layers. For Qwen2.5-1.5B, probes on the states after layers 14, 21, and 27 reach 0.541, 0.574, and 0.572, and a probe on the final normalized state, the vector the unembedding reads, reaches 0.803. Its AUC stays between 0.766 and 0.886 at every distance from 0 to 7 steps between the error and the question (Figure 1, right). A probe on the last block's output, before the final normalization, reaches 0.802, so the evidence becomes linearly readable within the last block. The Yes/No log-odds computed from the final state reach 0.544 overall and stay between 0.468 and 0.621 at every distance. Both models reject no corrupted derivation at the default threshold.

Readout geometry. Let μ_{clean} and μ_{corr} be the mean final states of clean and corrupted derivations, d the unit vector along $\mu_{\text{corr}} - \mu_{\text{clean}}$, and u the unit vector along $u' = \bar{w}_{\text{No}} - \bar{w}_{\text{Yes}}$, the mean No-token unembedding minus the mean Yes-token unembedding. The cosine between d and u is 0.069 at the final state (0.015 after layer 21). Projecting the final state onto d alone gives AUC 0.602, below the 0.803 of the full probe, so the evidence is spread across several directions. The spread of individual projections onto d is 2.57 times the gap between the class means. The unembedding reads the final state h linearly, so the No-minus-Yes score is $s = u'^{\top} h$, exact up to the log-sum-exp over token variants. For the same reason the direct-path contribution of the class difference, $(\mu_{\text{corr}} - \mu_{\text{clean}})^{\top} u' = 0.085$ nats, is close to the gap in mean verdict margin (5.553 for clean against 5.476 for corrupted); we report it as a check.

Decomposing the verdict score. A constant shift of the answer state would move clean and corrupted verdicts together and could not change the AUC, so we rescale each example's own

Table 1: Decomposition of the Yes/No verdict of Qwen2.5-1.5B on 192 short derivations: verdict AUC after the edit $h \leftarrow h + k\,p\,v$ at the answer position, with v the direction named in the row and $p = (h - \mu_{\text{clean}})^\top v$, applied at the final normalized state, where the score becomes $s + k\,c\,p$. $k = -1$ removes the component; $k = 10$ makes it 11 times its size. The corruption direction is estimated on all 1,280 derivations, so every entry is in-sample; Table 7 gives the cross-fitted version and earlier layers.

	$k = -1$	$k = 0$	$k = 10$	$k = 30$	$k = 100$
final state, corruption direction	0.558	0.663	0.787	0.804	0.810
final state, random direction	0.662	0.663	0.670	0.685	0.697

deviation along a direction:

$$h' = h + k\,p\,d, \qquad p = (h - \mu_{\text{clean}})^\top d,$$
$$s' = u'^\top h' = s + k\,c\,p, \qquad c = u'^\top d = 0.069\,\|u'\|.$$

The verdict score thus splits into a part along d, $c\,p$, and a part orthogonal to d. Removal ($k = -1$) keeps only the orthogonal part and a constant, and amplification ($k > 0$) scales up the part along d; as k grows the edited verdict ranks derivations by p alone, so its AUC tends to the AUC of p, a difference-of-means probe. A random direction has c near zero, so its edit should change little. We run the edit on 96 problems (192 derivations; arithmetic, depths 2 to 4) where the unmodified verdict has AUC 0.663, against 0.544 on all derivations. The direction is estimated on all 1,280 derivations, so Table 1 is an in-sample decomposition on cases where the verdict does better than average; Appendix E gives every condition.

At the final state, removal lowers the AUC from 0.663 to 0.558, and amplification raises it to 0.787 at $k = 10$ and 0.810 at $k = 100$, values that by the identity tend to the AUC of p on the same derivations, 0.814 (the full probe reaches 0.961 on them). The random direction moves the AUC by at most 0.034. Cross-fitting removes the in-sample estimate. With d and μ_{clean} estimated on the other four folds, removal takes the verdict AUC on all 1,280 derivations from 0.544 to 0.500, and amplification raises it to 0.585 at $k = 3$ and 0.596 at $k = 10$, against an out-of-fold AUC of p of 0.599; random directions move it by at most 0.004 (Table 7 in Appendix E).

Why this is more than a weak output classifier. The probe and the verdict read the same final normalized state. The verdict is one fixed linear readout of it, the No-minus-Yes row difference of the unembedding, and it realizes 15% of the discrimination the probe finds ($(0.544 - 0.5)/(0.803 - 0.5)$; 28% at 7B, Section 6). The part of the verdict score along the mean corruption direction d carries nearly all of its mean difference between clean and corrupted derivations. Removing that part takes the cross-fitted verdict to chance, and scaling it up moves the AUC toward that of p, 0.599, below the full probe's 0.803. These results describe the readout and leave open how the model computes it.

Earlier layers. Before the last block the mean direction carries little: after blocks 21 to 27 the cross-fitted p separates the classes with AUC 0.522 to 0.525. Rescaling that component at the output of block 21, 24, 26, or 27 changes the verdict AUC by -0.012 to $+0.022$, and random directions move it by less than 0.001. Adding the same vector only to the input of the final normalization, where no later block sees it, gives nearly the same change at blocks 26 and 27; at blocks 21 and 24 the part that passes through the later blocks changes the AUC by at most 0.016 (Table 7). All of these changes are small next to the 0.26 gap between the final-state probe and the verdict.

Controls. Four surface statistics of the derivation (token count, size of the final answer, largest value, number of negative values) separate corrupted from clean derivations with AUC 0.501 to 0.509, and a probe on all four reaches 0.505. The final-state probe scores the consistent twins of Section 3 like clean derivations. On a scale where 0 is the clean mean and 1 the corrupted mean, twins sit at 0.01 (95% CI -0.03 to 0.06), and the probe separates corrupted derivations from their twins as well as from their clean copies (AUC 0.788 and 0.789). The carry probe on the planted step does the same (twin against clean 0.504, corrupted against twin 0.892). Both probes read the inconsistent line, and the values a corrupted copy shares with its twin do not move them. Four worked examples in the prompt, two correct and two corrupted with their answers, leave the verdict at AUC

Table 2: The three stages in every model, on the same 1,280 derivations. Notice: derivation AUC from the most surprising step. Carry: last-layer probe for "an error has occurred so far" on the wrong step (t=0) and three steps later. Report probe: logistic probe on the answer-position state at layer $\frac{3}{4}L$ and at the final normalized state L. Yes/No: AUC of the model's verdict log-odds. Eff.: $(\text{Yes/No} - 0.5)/(L - 0.5)$, the share of the final-state probe's discrimination that the verdict realizes, given where that probe exceeds 0.7. cos: cosine between the corrupted-minus-clean mean difference of the final state and the No-minus-Yes unembedding direction. Qwen2.5-3B ran in fp16 and in fp32, Qwen2.5-7B with 4-bit NF4 weights, and every other model in fp32.

		Notice	Carry (last layer)		Report probe		Yes/No		
Model	Weights	AUC	t=0	t=3	$\frac{3}{4}L$	L	AUC	Eff.	cos
Qwen2.5-0.5B	fp32	0.872	0.764	0.521	0.545	0.545	0.507	–	0.034
Qwen2.5-1.5B	fp32	0.893	0.905	0.762	0.574	0.803	0.544	0.15	0.069
Qwen2.5-3B	fp16	0.920	0.843	0.584	0.580	0.601	0.539	–	0.244
Qwen2.5-3B	fp32	0.921	0.843	0.583	0.582	0.602	0.539	–	0.244
Qwen2.5-7B	4-bit	0.931	0.932	0.822	0.631	0.871	0.605	0.28	0.238
Llama-3.2-1B	fp32	0.859	0.837	0.629	0.534	0.595	0.503	–	0.026
Llama-3.2-3B	fp32	0.884	0.943	0.845	0.566	0.817	0.511	0.03	0.057
Gemma-2-2B	fp32	0.871	0.926	0.763	0.581	0.773	0.526	0.09	0.095

0.538 (change from zero-shot -0.006, 95% CI -0.020 to 0.007), while a probe on the final state of the same prompts reaches 0.851. Chain of thought does help. Asked to recompute every step and then give a verdict, the model rejects 52.3% of corrupted and 6.3% of clean derivations in a stratified subsample of 256 (verdict AUC 0.784, 95% CI 0.736 to 0.833, read at the verdict token), where the one-word verdict reaches 0.530 and the final-state probe 0.842. Recomputing each step in writing gives the verdict the model's own results to compare against, which the one-word verdict does not have. The base model Qwen2.5-1.5B, prompted without a chat template, shows the same split: surprisal reaches 0.891, the final-state probe 0.792, and the verdict 0.538.

6 SCALE

We repeat the stage measurements on Qwen2.5-3B-Instruct in fp16 and Qwen2.5-7B-Instruct with 4-bit NF4 weights, on the same 1,280 derivations, and rerun Qwen2.5-3B in fp32 with the code of Section 7. Weight precision differs across these runs, so this section tests whether the dissociation persists with size and makes no claim about scaling. Probes read the residual stream at layer $\frac{3}{4}L$ and the final normalized state at layer L. Table 2 (upper rows) collects the four sizes, and Figure 3 in Appendix C plots them.

Noticing improves with size. Across the four sizes, derivation AUC from surprisal rises from 0.872 at 0.5B to 0.931 at 7B, localization accuracy from 75.2% to 87.2%, and the mean surprisal gap at the planted step from 8.10 to 12.51 nats. Appendix C gives localization by position at 7B.

The verdict barely moves. Yes/No AUC is 0.507, 0.544, 0.539, and 0.605 from smallest to largest. The 7B model rejects 2.7% of corrupted derivations and no clean ones; the 3B model rejects 16.1% of corrupted derivations and 13.4% of clean ones. At 7B the final-state probe reaches 0.871 and stays above 0.76 at every distance from the error, so the answer state carries 0.27 AUC more than the verdict reports, the same margin as at 1.5B (0.26).

Carry and the final-state probe are not monotone in size. Both are high at 1.5B and 7B and low at 3B, where the carry probe three steps after the error reads 0.584 and the final-state probe 0.601. Rerun in fp32 with the same derivations and the code of Section 7, Qwen2.5-3B matches the fp16 run to within 0.003 AUC at every stage (0.583 and 0.602 for these two readouts), so precision does not explain the dip, and we make no claim about how carry scales. At 1.5B and 7B, where the answer state carries the evidence, the verdict realizes a small part of the probe's discrimination, 15% and 28%. The cosine between the corruption direction and the No-minus-Yes direction rises from 0.069 at 1.5B to 0.244 at 3B and 0.238 at 7B, so the cosine alone does not explain the weak verdict.

7 OTHER MODEL FAMILIES

We repeat the stage measurements on Llama-3.2-1B-Instruct, Llama-3.2-3B-Instruct (Grattafiori et al., 2024), and Gemma-2-2B-it (Gemma Team et al., 2024), all in fp32, with the same derivations, code, and probes (Table 2, lower rows). The same run repeats Qwen2.5-0.5B and 1.5B and reproduces their numbers.

Notice and report replicate. Surprisal separates corrupted from clean derivations with AUC 0.859 to 0.884 and finds the wrong step 71.3% to 79.8% of the time. Asked which step is wrong, the three models are right 22.3% to 28.1% of the time, against 22.9% for a uniform guess. The verdict reaches AUC 0.503 to 0.526, and its bias depends on the family. At the default threshold the Qwen2.5 models and Gemma-2-2B accept most derivations, while Llama-3.2-3B rejects every one and Llama-3.2-1B rejects 84% of clean and corrupted derivations alike. The answer also follows the wording of the question: asked instead whether the derivation contains a mistake, Gemma-2-2B says it does for 95% of derivations, clean or corrupted. Under that reversed question, and under a variant that first asks the model to check every step, the verdict AUC stays between 0.499 and 0.534 in all six fp32 models.

Carry and presence split by model. Gemma-2-2B and Llama-3.2-3B keep the evidence. Three steps after the error their carry probes read 0.763 and 0.845, and the final-state probe reaches 0.773 and 0.817, of which the verdict realizes 9% and 3%. Llama-3.2-1B keeps less: its carry probe falls from 0.837 on the wrong step to 0.629 three steps later, and its final-state probe reaches 0.595, between Qwen2.5-0.5B and 1.5B. In every family the next-token distribution notices the error and the verdict barely uses it. In the models of 1.5B parameters or more, the final-state probe finds the evidence the verdict leaves unused, with Qwen2.5-3B as the one exception, in fp16 and in fp32 (Section 6).

8 TRANSFER TO NATURAL ERRORS

The synthetic harness plants one clean error per derivation. Real solutions contain errors of every kind and phrasing, so we ask whether the notice signal alone localizes them. ProcessBench (Zheng et al., 2025) contains 3,400 competition and grade-school solutions, split into paragraphs, with the index of the first erroneous paragraph or -1 for correct solutions. Its metric is the harmonic mean of accuracy on erroneous solutions (predicting the exact first-error index) and accuracy on correct solutions (predicting -1).

Localizer. We feed the problem as the user turn and the solution as the assistant turn and read, for every paragraph, the mean surprisal of its tokens that contain a digit. The localizer flags the first paragraph whose score exceeds a threshold and predicts -1 if none does. We choose one statistic and one threshold on GSM8K and apply them unchanged to the other three subsets. The statistic is the best of eight candidates (mean, maximum, and top-three mean over all tokens, and the digit mean, each with and without subtracting the solution's median), and the threshold is the one with the best F1, the rule ProcessBench applies to the two Skywork process reward models; the other process reward models emit step labels and get no tuning. The GSM8K column of Table 3 is therefore in-sample only for our rows and the Skywork rows. Solutions longer than 1,600 tokens (8, 19, and 31 in MATH, OlympiadBench, and Omni-MATH) are counted as wrong. Restricting the average to digit-bearing tokens matters: the same localizer built on the mean over all tokens reaches an average F1 of 20.3 (Appendix D).

Result. Table 3 gives the numbers. The digit surprisal of Qwen2.5-Math-1.5B-Instruct (Yang et al., 2024) reaches an average F1 of 42.9, above Math-Shepherd-PRM-7B (31.5; Wang et al., 2024) and Skywork-PRM-1.5B (36.4), close to Skywork-PRM-7B (42.1), and below two 7B models trained on step labels, Qwen2.5-Math-7B-PRM800K (56.5) and Qwen2.5-Math-PRM-7B (73.5; Zhang et al., 2025c). The general Qwen2.5-1.5B-Instruct reaches 36.7. Both beat prompted 7B critics. On the three subsets not used for selection, the math model averages 38.8, against 21.0 to 52.5 for the process reward models in the benchmark paper and 70.5 for Qwen2.5-Math-PRM-7B. On the two hardest subsets its surprisal (37.5 and 33.4) is above every process reward model in the benchmark paper except Qwen2.5-Math-7B-PRM800K.

Table 3: ProcessBench F1 ($\times 100$). Published rows are copied from Zheng et al. (2025), except the Qwen2.5-Math-PRM-7B row (†), which is from Zhang et al. (2025c); the prompted critics use majority voting over eight samples. Our rows involve no training: one statistic and one threshold are chosen on the 400 GSM8K solutions. Among the published process reward models, only the two Skywork models have a threshold tuned on GSM8K; the others emit step labels directly. Avg.$_3$ averages MATH, OlympiadBench, and Omni-MATH, the subsets not used for our selection.

Method	GSM8K	MATH	OlympiadBench	Omni-MATH	Avg.	Avg.$_3$
Trained process reward models (reported by Zheng et al. (2025), except †)						
Math-Shepherd-PRM-7B	47.9	29.5	24.8	23.8	31.5	26.0
RLHFlow-PRM-Mistral-8B	50.4	33.4	13.8	15.8	28.4	21.0
RLHFlow-PRM-Deepseek-8B	38.8	33.8	16.9	16.9	26.6	22.5
Skywork-PRM-1.5B	59.0	48.0	19.3	19.2	36.4	28.8
Skywork-PRM-7B	70.8	53.6	22.9	21.0	42.1	32.5
Qwen2.5-Math-7B-PRM800K	68.2	62.6	50.7	44.3	56.5	52.5
Qwen2.5-Math-PRM-7B†	82.4	77.6	67.5	66.3	73.5	70.5
Prompted critics, majority vote over 8 samples (reported by Zheng et al. (2025))						
Qwen2.5-Math-7B-Instruct	26.8	25.7	14.2	12.7	19.9	17.5
Qwen2.5-7B-Instruct	36.5	36.6	29.7	27.4	32.6	31.2
Ours: the scoring model's own digit surprisal, no process labels						
Qwen2.5-1.5B-Instruct	49.8	39.2	28.0	29.9	36.7	32.4
Qwen2.5-Math-1.5B-Instruct	55.2	45.5	37.5	33.4	42.9	38.8
Same models, one-token first-error index (solutions with at most 10 steps)						
Qwen2.5-1.5B-Instruct	0.0	0.5	4.7	7.4	3.2	4.2
Qwen2.5-Math-1.5B-Instruct	0.0	0.0	0.0	2.8	0.7	0.9

The same model's verbal readout. Asked for the index of the earliest wrong paragraph as a single token, the same two models reach F1 of 0.7 and 3.2; the benchmark's critic protocol instead generates a review and votes over eight samples (Appendix D). They almost never answer -1 (0% to 4.4% of correct solutions) and they point early: on erroneous solutions the mean predicted position is 0.19 to 0.28 of the way through the solution, against a true mean of 0.39 to 0.44. The dissociation between surprisal and verbal localization from Section 3 carries over to natural errors. The next-token distribution registers where a solution goes wrong, and the model's answer to the same question barely does.

Caveats. Qwen2.5-1.5B-Instruct is one of the twelve models that generated ProcessBench solutions (Zheng et al., 2025), so it scores some of its own text and we treat its row as secondary; Qwen2.5-Math-1.5B-Instruct is not among the generators, and its row is the one we compare with the trained models. Two generators, Qwen2.5-Math-7B-Instruct and Qwen2.5-Math-72B-Instruct, come from the same model family as our scorer, so the scorer may favor their solutions. On MATH, OlympiadBench, and Omni-MATH the most surprising paragraph also sits later than the true first error (mean relative position 0.54 to 0.58 against 0.39 to 0.43), so the two readouts err in opposite directions.

9 Related work

Probability and prompting. Direct probability measurements often reveal knowledge that prompted judgments miss (Hu & Levy, 2023), most of all in small models (Hu & Frank, 2024). Prompted judgments about a model's own knowledge of language show little privileged access to its probabilities (Song et al., 2025b). Language models often generate answers they fail to validate (Li et al., 2024; West et al., 2024), and this generation-verification gap largely governs self-improvement (Song et al., 2025c). Asked to locate a reasoning error, models struggle, though they can fix it once told where it is (Tyen et al., 2024). Our verdict gap is an instance of this pattern in multi-step arithmetic, and we follow the evidence from the written step to the answer.

Verification and error detection. Process reward models and step-level verifiers are trained to judge reasoning traces (Lightman et al., 2024; Wang et al., 2024; Zhang et al., 2025b;c), and benchmarks measure first-error identification (Zheng et al., 2025), fine-grained step-level error

detection (Song et al., 2025a), and response-level error detection (Kamoi et al., 2024). PRMBench reports that some trained process reward models improve as the erroneous step moves later (Song et al., 2025a). Self-correction without external feedback is unreliable (Huang et al., 2024; Kumar et al., 2025). Gadetsky et al. (2026) score steps from next-token probabilities of correctness labels to build process rewards without annotation, and an implicit process reward model can be trained from outcome labels alone (Yuan et al., 2025); our notice readout scores the written content itself and is used as a diagnostic.

Internal signals of mistakes. Hidden states encode truthfulness and correctness beyond what models say (Azaria & Mitchell, 2023; Burns et al., 2023; Marks & Tegmark, 2024; Orgad et al., 2025), including factual knowledge a model does not express (Gekhman et al., 2025). Probes on hidden states detect when a model has made a reasoning mistake (Ye et al., 2025), whether a reasoning model's intermediate answer is correct (Zhang et al., 2025a), and whether a chain of thought will succeed before it is finished (Afzal et al., 2025). Attribution graphs of reasoning steps carry error signatures that support targeted corrections (Zhao et al., 2026), and models can report and control their activations along some directions more easily than others (Ji-An et al., 2025). For arithmetic, Sun et al. (2025) decode predicted and correct answers from hidden states and use probes to trigger re-prompting, and Bertolazzi et al. (2025) find that small models validate single-step arithmetic with consistency heads in middle layers, before the arithmetic result is fully encoded. In our setting the error is planted in text the model reads, so its position and size are fixed by construction.

Interventions. Inference-time intervention shifts activations along probe directions to change a model's answers (Li et al., 2023), and activation steering adds the same fixed vector to every example (Turner et al., 2023). Our corruption direction is a difference of class means, as in Marks & Tegmark (2024), and our edit rescales each example's own component. At the final normalized state such an edit changes the logits linearly, so we report it as a decomposition of the readout.

10 Limitations and conclusion

The problems are arithmetic chains with one planted error; natural reasoning errors are messier, which Section 8 tests. The notice statistic is next-token surprisal, which responds to any unexpected token; the consistent twins of Section 3 show that in our harness the written numbers alone do not produce the gap. The seven models come from three families and range from 0.5B to 7B parameters; the Qwen2.5-7B run used 4-bit weights, Qwen2.5-3B ran in both fp16 and fp32, and the readout decomposition covers only Qwen2.5-1.5B. Probe accuracy is a lower bound on what a readout could extract. The twins and the surface baselines rule out the written numbers and the surface of the text as what the probes read, in the spirit of a control task (Hewitt & Liang, 2019), but they do not identify the feature of the inconsistency that the probe uses, and probe accuracy says nothing about whether the model would use the evidence without intervention. A mean difference captures only part of what the full probe uses. We tested three phrasings of the verdict question on six models, and four-shot prompting and chain of thought on Qwen2.5-1.5B; other ways of eliciting a verdict may do better. Whether the carry stage depends on architectural choices such as the positional encoding is untested.

Conclusion. In all seven models, from three families, the next-token distribution registers a wrong step when it is read. In Qwen2.5-1.5B and 7B, Gemma-2-2B, and Llama-3.2-3B the state at the answer position still encodes it, and the one-word verdict uses little of that evidence. For these models the bottleneck is the report. In Qwen2.5-1.5B, four worked examples leave the verdict near chance, and so does the base model without instruction tuning, while asking the model to recompute each step in writing lets it reject about half of the corrupted derivations. Qwen2.5-0.5B, Llama-3.2-1B, and Qwen2.5-3B lose most of the evidence earlier. A verifier built on a model's own step-level surprisal starts from a signal the model already computes. On ProcessBench this signal from a 1.5B model is competitive with early process reward models and uses no process labels. We have not tested a linear readout of the answer state as a verifier on natural errors. Training aimed at the readout itself is the next test.

REPRODUCIBILITY STATEMENT

All problems are generated from a fixed seed with no external data; the only downloads are public model checkpoints. Every number in this paper is read from result files written by the released code, and every figure is drawn from those files. The code runs on a single T4 GPU.

AUTHOR CONTRIBUTION AND AI ASSISTANCE STATEMENT

The authors developed the core research goals, designed the experimental framework, and authored the manuscript. A generative AI assistant (Claude, Anthropic) was used throughout the project for code scripting, figure and table formatting, literature search, and editing. For the revision it also wrote and ran the analysis code for the control, model-family, and precision experiments and drafted the text that reports them. The authors reviewed that code, its results, and the text, and take full responsibility for the contents of this work.

REFERENCES

Anum Afzal, Florian Matthes, Gal Chechik, and Yftah Ziser. Knowing before saying: LLM representations encode information about chain-of-thought success before completion. *arXiv preprint arXiv:2505.24362*, 2025. URL https://arxiv.org/abs/2505.24362.

Amos Azaria and Tom Mitchell. The internal state of an LLM knows when it's lying. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, Singapore, 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-emnlp.68. URL https://aclanthology.org/2023.findings-emnlp.68/.

Leonardo Bertolazzi, Philipp Mondorf, Barbara Plank, and Raffaella Bernardi. The Validation Gap: A Mechanistic Analysis of How Language Models Compute Arithmetic but Fail to Validate It. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 29387–29424, 2025.

Collin Burns, Haotian Ye, Dan Klein, and Jacob Steinhardt. Discovering Latent Knowledge in Language Models Without Supervision. In *International Conference on Learning Representations*, 2023.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Łukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training Verifiers to Solve Math Word Problems. *arXiv preprint arXiv:2110.14168*, 2021.

Artyom Gadetsky, Maxim Kodryan, Siba Smarak Panigrahi, Hang Guo, and Maria Brbic. Unsupervised Process Reward Models. *arXiv preprint arXiv:2605.10158*, 2026.

Zorik Gekhman, Eyal Ben-David, Hadas Orgad, Eran Ofek, Yonatan Belinkov, Idan Szpektor, Jonathan Herzig, and Roi Reichart. Inside-out: Hidden factual knowledge in LLMs. In *Second Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=f7GG1MbsSM.

Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, et al. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*, 2024. URL https://arxiv.org/abs/2408.00118.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, et al. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024. URL https://arxiv.org/abs/2407.21783.

John Hewitt and Percy Liang. Designing and interpreting probes with control tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, Hong Kong, China, 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1275. URL https://aclanthology.org/D19-1275/.

Jennifer Hu and Michael C. Frank. Auxiliary task demands mask the capabilities of smaller language models. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=U5BUzSn4tD.

Jennifer Hu and Roger Levy. Prompting is not a substitute for probability measurements in large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, Singapore, 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.306. URL https://aclanthology.org/2023.emnlp-main.306/.

Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Wei Yu, Xinying Song, and Denny Zhou. Large Language Models Cannot Self-Correct Reasoning Yet. In *International Conference on Learning Representations*, 2024.

Li Ji-An, Hua-Dong Xiong, Robert C. Wilson, Marcelo G. Mattar, and Marcus K. Benna. Language models are capable of metacognitive monitoring and control of their internal activations. In *Advances in Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=qTXlFwlggv.

Ryo Kamoi, Sarkar Snigdha Sarathi Das, Renze Lou, Jihyun Janice Ahn, Yilun Zhao, Xiaoxin Lu, Nan Zhang, Yusen Zhang, Ranran Haoran Zhang, Sujeeth Reddy Vummanthala, Salika Dave, Shaobo Qin, Arman Cohan, Wenpeng Yin, and Rui Zhang. Evaluating LLMs at Detecting Errors in LLM Responses. In *Conference on Language Modeling*, 2024.

Aviral Kumar, Vincent Zhuang, Rishabh Agarwal, Yi Su, John D Co-Reyes, Avi Singh, Kate Baumli, Shariq Iqbal, Colton Bishop, Rebecca Roelofs, Lei M Zhang, Kay McKinney, Disha Shrivastava, Cosmin Paduraru, George Tucker, Doina Precup, Feryal Behbahani, and Aleksandra Faust. Training Language Models to Self-Correct via Reinforcement Learning. In *International Conference on Learning Representations*, 2025.

Kenneth Li, Oam Patel, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. Inference-time intervention: Eliciting truthful answers from a language model. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://openreview.net/forum?id=aLLuYpn83y.

Xiang Lisa Li, Vaishnavi Shrivastava, Siyan Li, Tatsunori Hashimoto, and Percy Liang. Benchmarking and improving generator-validator consistency of language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=phBS6YpTzC.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's Verify Step by Step. In *International Conference on Learning Representations*, 2024.

Samuel Marks and Max Tegmark. The Geometry of Truth: Emergent Linear Structure in Large Language Model Representations of True/False Datasets. In *Conference on Language Modeling*, 2024.

Hadas Orgad, Michael Toker, Zorik Gekhman, Roi Reichart, Idan Szpektor, Hadas Kotek, and Yonatan Belinkov. LLMs Know More Than They Show: On the Intrinsic Representation of LLM Hallucinations. In *International Conference on Learning Representations*, 2025.

Qwen, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 Technical Report. *arXiv preprint arXiv:2412.15115*, 2024.

Mingyang Song, Zhaochen Su, Xiaoye Qu, Jiawei Zhou, and Yu Cheng. PRMBench: A Fine-grained and Challenging Benchmark for Process-Level Reward Models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 25299–25346, 2025a.

Siyuan Song, Jennifer Hu, and Kyle Mahowald. Language models fail to introspect about their knowledge of language. In *Second Conference on Language Modeling*, 2025b. URL https://openreview.net/forum?id=AivRDOFi5H.

Yuda Song, Hanlin Zhang, Carson Eisenach, Sham Kakade, Dean Foster, and Udaya Ghai. Mind the gap: Examining the self-improvement capabilities of large language models. In *The Thirteenth International Conference on Learning Representations*, 2025c. URL https://arxiv.org/abs/2412.02674.

Yucheng Sun, Alessandro Stolfo, and Mrinmaya Sachan. Probing for Arithmetic Errors in Language Models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 8111–8128, 2025.

Alexander Matt Turner, Lisa Thiergart, Gavin Leech, David Udell, Juan J. Vazquez, Ulisse Mini, and Monte MacDiarmid. Steering Language Models With Activation Engineering. *arXiv preprint arXiv:2308.10248*, 2023.

Gladys Tyen, Hassan Mansoor, Victor Carbune, Peter Chen, and Tony Mak. LLMs cannot find reasoning errors, but can correct them given the error location. In *Findings of the Association for Computational Linguistics: ACL 2024*, Bangkok, Thailand, 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-acl.826. URL https://aclanthology.org/2024.findings-acl.826/.

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. Math-shepherd: Verify and reinforce LLMs step-by-step without human annotations. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Bangkok, Thailand, 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.510. URL https://aclanthology.org/2024.acl-long.510/.

Peter West, Ximing Lu, Nouha Dziri, Faeze Brahman, Linjie Li, Jena D. Hwang, Liwei Jiang, Jillian Fisher, Abhilasha Ravichander, Khyathi Chandu, Benjamin Newman, Pang Wei Koh, Allyson Ettinger, and Yejin Choi. The generative AI paradox: "what it can create, it may not understand". In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=CF8H8MS5P8.

An Yang, Beichen Zhang, Binyuan Hui, Bofei Gao, Bowen Yu, Chengpeng Li, Dayiheng Liu, Jianhong Tu, Jingren Zhou, Junyang Lin, Keming Lu, Mingfeng Xue, Runji Lin, Tianyu Liu, Xingzhang Ren, and Zhenru Zhang. Qwen2.5-Math technical report: Toward mathematical expert model via self-improvement. *arXiv preprint arXiv:2409.12122*, 2024. URL https://arxiv.org/abs/2409.12122.

Tian Ye, Zicheng Xu, Yuanzhi Li, and Zeyuan Allen-Zhu. Physics of language models: Part 2.2, how to learn from mistakes on grade-school math problems. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=zpDGwcmMV4.

Lifan Yuan, Wendi Li, Huayu Chen, Ganqu Cui, Ning Ding, Kaiyan Zhang, Bowen Zhou, Zhiyuan Liu, and Hao Peng. Free process rewards without process labels. In *Proceedings of the 42nd International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=8ThnPFhGm8.

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. In *Second Conference on Language Modeling*, 2025a. URL https://arxiv.org/abs/2504.05419.

Lunjun Zhang, Arian Hosseini, Hritik Bansal, Mehran Kazemi, Aviral Kumar, and Rishabh Agarwal. Generative Verifiers: Reward Modeling as Next-Token Prediction. In *International Conference on Learning Representations*, 2025b.

Zhenru Zhang, Chujie Zheng, Yangzhen Wu, Beichen Zhang, Runji Lin, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. The lessons of developing process reward models in mathematical reasoning. *arXiv preprint arXiv:2501.07301*, 2025c. URL https://arxiv.org/abs/2501.07301.

Zheng Zhao, Yeskendir Koishekenov, Xianjun Yang, Naila Murray, and Nicola Cancedda. Verifying chain-of-thought reasoning via its computational graph. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=CxiNICq0Rr.

Chujie Zheng, Zhenru Zhang, Beichen Zhang, Runji Lin, Keming Lu, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. ProcessBench: Identifying Process Errors in Mathematical Reasoning. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1009–1024, 2025.

A HARNESS DETAILS

Generator. All runs use the same seeded generator (seed 0), so every model sees identical problems, error positions, and error sizes. For each of two families and each depth from 2 to 9 we draw 40 problems, giving 640 problems and 1,280 derivations.

- *Arithmetic.* The start value is uniform on $\{2, \ldots, 50\}$. Each step picks an operation from $\{+, -, \times, +, -\}$ uniformly, so addition and subtraction are twice as likely as multiplication. Multipliers are uniform on $\{2, \ldots, 9\}$ and addends on $\{2, \ldots, 40\}$. If the running value exceeds 400 in absolute value, a drawn multiplication is replaced by an addition or subtraction.
- *Modular.* The modulus is uniform on $\{7, 11, 13\}$, the start value on $\{0, \ldots, m-1\}$, the operation on $\{+, \times\}$, and the operand on $\{2, \ldots, m-1\}$. Every result is reduced modulo m.

Planted error. The error position is uniform on $\{1, \ldots, d\}$ for depth d. Even-indexed problems in each cell receive a *subtle* error, an offset drawn from $\{\pm 1, \pm 2, \pm 3\}$; odd-indexed problems receive a *gross* error, an offset of $\pm u$ with u uniform on $\{10, \ldots, 60\}$. In the modular family the offset is reduced modulo m, so only the arithmetic family has a meaningful size distinction. Offsets that leave the result unchanged are redrawn. Every later step is recomputed from the wrong value, so the corrupted derivation contains exactly one locally inconsistent line and a final answer consistent with that line.

Prompt. The user turn is the problem statement, a blank line, the derivation, a blank line, and the question "Is every step of this derivation arithmetically correct? Answer with one word: Yes or No." It is wrapped in the model's chat template with the generation prompt, and the verdict is read from the next-token distribution. The written result of each step is located by character offsets in the prompt and mapped to the tokens that overlap it; step surprisal is the summed negative log-probability of those tokens under teacher forcing.

Verbal localization. The localization numbers in Sections 3 and 7 come from the model-family run (Appendix F), in fp32, on the same 640 corrupted derivations; an earlier fp16 run of the two Qwen2.5 models gave 23.6% and 23.4%. The prompt replaces the verdict question with "Exactly one step of this derivation is arithmetically wrong. Which step number is wrong? Answer with just the number." and the prediction is the step number whose token has the highest next-token probability.

Examples. Problem 3 (arithmetic, depth 2, gross error at step 1) and problem 360 (modular, depth 3, subtle error at step 1):

Clean

```
Start with 8.  Then:  add 23; add
37.
Step 1:  8 + 23 = 31
Step 2:  31 + 37 = 68
Final answer:  68
```

Corrupted

```
Step 1:  8 + 23 = 68
Step 2:  68 + 37 = 105
Final answer:  105
```

Clean (modulus 13)

```
Step 1:  (8 + 5) mod 13 = 0
Step 2:  (0 * 12) mod 13 = 0
Step 3:  (0 * 4) mod 13 = 0
Final answer:  0
```

Corrupted

```
Step 1:  (8 + 5) mod 13 = 10
Step 2:  (10 * 12) mod 13 = 3
Step 3:  (3 * 4) mod 13 = 12
Final answer:  12
```

B PROBE DETAILS

Protocol. Every probe is an ℓ_2-regularized logistic regression ($C = 0.05$, standardized features, up to 2,000 iterations). Scores are out of fold from five folds grouped by problem, so the clean and corrupted copies of a problem are always in the same fold, and AUC is computed on the pooled out-of-fold scores. The model-family and precision runs (Appendix F) also refit the final-state probe with $C \in \{0.01, 0.05, 0.2\}$ and three assignments of problems to folds. Across these nine fits the AUC of each model spans at most 0.031: Qwen2.5-1.5B ranges from 0.790 to 0.805, Llama-3.2-3B

from 0.794 to 0.817, Gemma-2-2B from 0.758 to 0.789, and Qwen2.5-3B from 0.589 to 0.609, so the dip at 3B does not come from the probe's settings.

Positions and layers. The *readout* position is the last prompt token, where the model predicts its verdict. The *step* position is the last token of each step's written result. Layer l refers to `hidden_states[`l`]`: the residual stream after decoder block l for $l < L$, and the output of the final normalization for $l = L$, which is the vector the unembedding reads. The harness run probes layers at $\frac{1}{4}$, $\frac{1}{2}$, $\frac{3}{4}$ and all of depth; the scale runs probe $\frac{3}{4}L$ and L.

Carry probe. At every step position of every derivation the label is 1 if a wrong step has occurred at or before that step. To measure carry at offset t we take the positives exactly t steps after the planted error and, as negatives, all clean-derivation positions with the same step indices, so the comparison is at matched position.

Readout probe by distance. Table 4 gives the final-state probe and the verdict at each distance between the error and the question.

Table 4: AUC at the answer position by the number of steps between the planted error and the question. P: probe on the final normalized state. V: Yes/No verdict. Pairs: corrupted derivations at that distance. All models in fp32 except Qwen2.5-7B (4-bit weights).

	0.5B		1.5B		3B		7B		
Distance	P	V	P	V	P	V	P	V	Pairs
0	0.536	0.523	0.801	0.561	0.644	0.552	0.918	0.625	149
1	0.555	0.493	0.823	0.544	0.622	0.561	0.889	0.634	140
2	0.566	0.491	0.799	0.538	0.638	0.532	0.886	0.620	106
3	0.539	0.513	0.809	0.561	0.542	0.522	0.825	0.553	81
4	0.585	0.494	0.766	0.566	0.561	0.482	0.822	0.588	55
5	0.500	0.504	0.773	0.621	0.535	0.578	0.816	0.587	46
6	0.499	0.545	0.795	0.468	0.508	0.474	0.769	0.498	29
7	0.523	0.552	0.886	0.602	0.622	0.586	0.869	0.586	25

C ADDITIONAL STAGE RESULTS

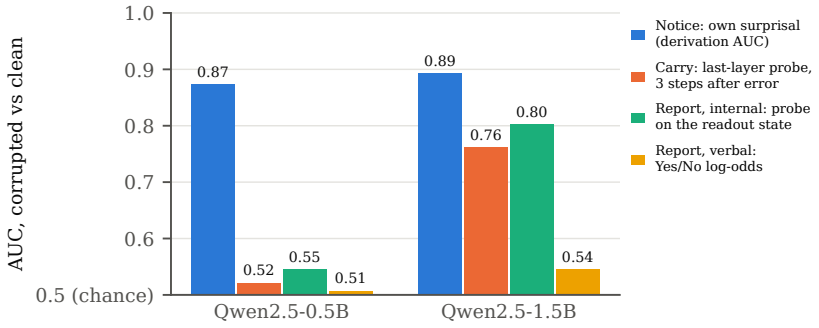

Figure 2: The three stages on the harness. Notice uses the model's own surprisal; carry is a last-layer probe three steps after the error, so it uses only derivations with at least three steps after the error; the two report readouts are a probe on the final normalized state at the answer position and the model's Yes/No log-odds from that same state.

Stage summary. Figure 2 shows the three stages side by side.

Surprisal after the error. After the planted step, surprisal returns to the clean-copy level within one step: the mean excess at the next step is 0.03 nats for Qwen2.5-0.5B and 0.16 for Qwen2.5-1.5B.

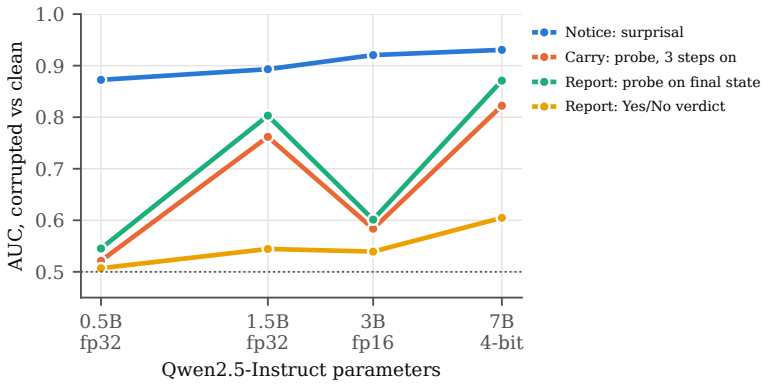

Figure 3: Stage AUCs for the four Qwen2.5 sizes of Section 6. Tick labels give the weight precision of each run; the 3B point is the fp16 run; Table 2 adds the fp32 rerun. The Yes/No verdict stays below 0.61 at every size while surprisal separates corrupted from clean derivations at 0.87 or better.

Localization by position at 7B. The 7B model finds the wrong step 94.8% of the time when it sits in the first quarter of the derivation and 83.2% of the time when it sits in the last quarter.

Carry at earlier layers. Probes on earlier layers carry less than the last layer: 0.637 to 0.682 at the error step for layers at one half and three quarters of depth.

D PROCESSBENCH DETAILS

Data. GSM8K, MATH, OlympiadBench, and Omni-MATH contain 400, 1,000, 1,000, and 1,000 solutions, of which 207, 594, 661, and 759 contain an error; mean solution lengths are 5.2, 6.5, 8.8, and 8.3 paragraphs. Both scoring models run in fp16.

Statistics and protocols. Table 5 gives the average F1 of every candidate statistic under both protocols. *GSM8K threshold* is the protocol of Section 8, the rule ProcessBench applies to the Skywork process reward models: the threshold with the best F1 on all of GSM8K is applied to all four subsets, and over-length solutions count as wrong. *Per-subset* picks the threshold on a random 25% of each subset and scores the other 75%, excluding over-length solutions; under it, Qwen2.5-Math-1.5B-Instruct and Qwen2.5-1.5B-Instruct reach average F1 of 42.2 and 36.8. Averaging over all tokens of a paragraph gives about half the F1 of averaging over digit-bearing tokens only; subtracting the solution's median changes little. The selected thresholds are 0.138 nats (Qwen2.5-Math-1.5B-Instruct) and 0.058 nats (Qwen2.5-1.5B-Instruct).

Table 5: ProcessBench average F1 ($\times 100$) of each paragraph statistic under the two protocols.

	Qwen2.5-Math-1.5B-Instruct		Qwen2.5-1.5B-Instruct	
Statistic	GSM8K threshold	per-subset	GSM8K threshold	per-subset
mean	20.3	22.4	20.9	22.6
mean, minus median	17.4	18.1	19.9	21.0
maximum	18.9	17.4	18.6	18.7
maximum, minus median	16.5	14.6	15.9	15.7
top-3 mean	22.0	20.9	21.8	21.7
top-3 mean, minus median	17.3	15.3	17.0	16.8
digit mean	42.9	42.2	36.7	36.8
digit mean, minus median	40.9	40.7	36.4	36.8

Excluded run. We also prompted Qwen2.5-Math-1.5B-Instruct as a critic with the ProcessBench prompt on GSM8K and MATH. With a 96-token generation budget the model rarely finished its

Table 6: Accuracy ($\times 100$) of the digit-mean localizer on erroneous solutions (exact first-error index) and on correct solutions (predicting no error), per-subset protocol, held-out 75%.

Model		GSM8K	MATH	OlympiadBench	Omni-MATH
Qwen2.5-Math-1.5B	erroneous	45.1	36.2	26.2	22.7
	correct	63.0	64.4	67.6	55.5
Qwen2.5-1.5B	erroneous	42.0	32.9	24.5	20.9
	correct	34.8	61.9	63.4	59.4

review and boxed an index, so the parsed answers are mostly incidental numbers from the review text. We report no number from this run.

E GAIN EDIT DETAILS

The gain edit of Section 5 uses Qwen2.5-1.5B-Instruct in fp32 on problems 0 to 95 of the harness, which are the 40 arithmetic problems of depth 2, the 40 of depth 3, and the first 16 of depth 4 (192 derivations). The corruption direction d and the clean mean μ_{clean} are computed on all 1,280 derivations at the same layer, and the hook rescales only the state at the answer position. The random direction is a normalized draw from an isotropic Gaussian (seed 0), drawn separately for each layer. Table 8 lists every condition with the mean Yes-minus-No margin and the rejection rates at the default threshold (margin at most zero). Because the direction is estimated on a set that contains the 96 edited problems, these numbers are an in-sample estimate of the effect.

After layer 21. This edit passes through the last seven decoder blocks and the final normalization before the unembedding reads it, so the identity of Section 5 does not fix its effect. Removal leaves the AUC at 0.664 and amplification lowers it, to 0.632 at $k = 10$ and 0.629 at $k = 30$; at $k = 100$ the ordering flips (AUC 0.426), with 36.5% of clean and 28.1% of corrupted derivations rejected. The probe after layer 21 reads only 0.574 and the cosine between d and u there is 0.015, so the direction estimated at this layer is noisy and the test says little either way.

Cross-fitted edits. The control run (Appendix F) repeats the decomposition on all 1,280 derivations with the direction and the clean mean estimated on the other four folds (grouped by problem), and adds the same edit after blocks 21, 24, 26, and 27, either at that block's output or at the input of the final normalization only. Table 7 gives the results. Read through the unembedding, the final-state direction lowers 'Yes' most (0.157 logits per unit) and lowers 'No' (0.088) about as much as the other Yes variants (0.079 to 0.092); the tokens it raises most are ordinal words such as 'third', 'fourth', and 'second'.

Table 7: Cross-fitted edits on Qwen2.5-1.5B-Instruct, all 1,280 derivations: verdict AUC after rescaling the component along the cross-fitted mean corruption direction (unedited: 0.544). Final state: computed exactly from the stored states. Later rows: added at the output of block l, where every later block sees it, and, after the slash, at the input of the final normalization only, with the added projection clipped to ± 3 standard deviations. p: out-of-fold AUC of the projection itself. Random directions move every entry by at most 0.004.

Site	p	$k = -1$	$k = 3$	$k = 10$
Final state	0.599	0.500	0.585	0.596
After block 21	0.522	0.561 / 0.545	0.532 / 0.543	0.537 / 0.542
After block 24	0.523	0.564 / 0.551	0.533 / 0.537	0.539 / 0.540
After block 26	0.524	0.566 / 0.561	0.533 / 0.534	0.537 / 0.537
After block 27	0.525	0.563 / 0.562	0.534 / 0.534	0.537 / 0.537

Table 8: All gain-edit conditions on Qwen2.5-1.5B-Instruct, 192 derivations. The direction is estimated on a set that includes them, so every entry is in-sample.

Layer	Direction	k	AUC	margin, clean	margin, corrupted	reject corrupted	reject clean
final state	corruption	-1	0.558	5.66	5.58	0.0%	0.0%
final state	corruption	0	0.663	5.95	5.68	0.0%	0.0%
final state	corruption	10	0.787	8.82	6.76	0.0%	0.0%
final state	corruption	30	0.804	14.57	8.91	5.2%	0.0%
final state	corruption	100	0.810	34.69	16.17	18.8%	1.0%
final state	random	-1	0.662	5.94	5.68	0.0%	0.0%
final state	random	0	0.663	5.95	5.68	0.0%	0.0%
final state	random	10	0.670	6.00	5.72	0.0%	0.0%
final state	random	30	0.685	6.10	5.80	0.0%	0.0%
final state	random	100	0.697	6.47	6.06	0.0%	0.0%
after layer 21	corruption	-1	0.664	5.66	5.52	0.0%	0.0%
after layer 21	corruption	0	0.663	5.95	5.68	0.0%	0.0%
after layer 21	corruption	10	0.632	8.71	7.42	1.0%	0.0%
after layer 21	corruption	30	0.629	11.51	9.77	14.6%	11.5%
after layer 21	corruption	100	0.426	3.14	4.48	28.1%	36.5%
after layer 21	random	-1	0.662	5.94	5.68	0.0%	0.0%
after layer 21	random	0	0.663	5.95	5.68	0.0%	0.0%
after layer 21	random	10	0.662	5.96	5.69	0.0%	0.0%
after layer 21	random	30	0.661	5.98	5.72	0.0%	0.0%
after layer 21	random	100	0.659	6.10	5.82	0.0%	0.0%

F COMPUTE

Every language-model run used one NVIDIA Tesla T4 (16 GB) on Google Colab with PyTorch 2.11 and Transformers 5.16.1, SDPA attention (eager for Gemma-2, whose attention uses logit softcapping), and seed 0. Llama-3.2 and Gemma-2 weights come from public mirrors of the official checkpoints that do not require a license token. Table 9 lists the runs.

Table 9: Runs, weight precision, and wall-clock time on one T4.

Run	Models and precision	Minutes
Harness stages (Secs. 3 to 5)	Qwen2.5-0.5B and 1.5B-Instruct, fp32	9.8
Direction and steering (Sec. 5)	Qwen2.5-1.5B-Instruct, fp32	12.2
Scale (Sec. 6)	Qwen2.5-3B-Instruct, fp16; Qwen2.5-7B-Instruct, 4-bit NF4	21.2
Verbal localization, first run	Qwen2.5-0.5B and 1.5B-Instruct, fp16 (also solves each problem with and without chain of thought)	14.4
Model families, twins, paraphrases (Secs. 3, 7)	Qwen2.5-0.5B and 1.5B, Llama-3.2-1B and 3B, Gemma-2-2B (Instruct), fp32	128.4
ProcessBench (Sec. 8)	Qwen2.5-Math-1.5B-Instruct and Qwen2.5-1.5B-Instruct, fp16	62.4
Controls (Sec. 5)	Qwen2.5-1.5B-Instruct and Qwen2.5-1.5B (base), fp32	36.6
3B precision control (Sec. 6)	Qwen2.5-3B-Instruct, fp32	44.9

A first attempt at the harness run in fp16 with eager attention produced non-finite activations in Qwen2.5-1.5B; all harness runs therefore use fp32 for the two small models, and the scale runs check every loaded model for non-finite logits and hidden states before use.