RLVR AS PROMPT AUGMENTATION REINFORCES BOTH CORRECT AND INCORRECT REASONING TO THE CORRECT ANSWER

Anonymous authors
Paper under double-blind review

ABSTRACT

Reinforcement Learning with Verifiable Rewards (RLVR)-based post-training of Large Language Models (LLMs) has been shown to improve accuracy on reasoning tasks and continues to attract significant attention. While RLVR is known to improve solution accuracy, there has been little agreement on its effect on the so-called reasoning traces–with previous investigations coming to differing conclusions. We argue that outcome rewards cannot distinguish between reasoning from any other scaffold that leads to final answer correctness. To support our claims, we show that RLVR as learning prompt-augmentation conditioned on solution correctness, where a CoT is reinforced when it raises the probability that the model produces a correct final answer independent of whether CoT is correct or not. The objective is indifferent to every other property of the CoT including semantics, interpretability and faithfulness. We derive testable predictions and evaluate the prior results in a formally verifiable domain and test on GSM8K and two of its variants, GSM8K-Symbolic-P1 and GSM8K-Symbolic-P2. For GSM8K, where trace validity cannot be reliably established, we measure trace coherence, defined by the absence of errors. We compute α, the probability of a correct final answer given a coherent trace, and β, the probability given an incoherent one, for the base model and after RLVR training. Across all settings β rises without exception, contradicting the claim that RLVR implicitly incentivizes correct reasoning. Thus RLVR learns prompt augmentations conditioned on final answer correctness: the objective does not specifically upweight valid reasoning, but any CoT that increases final-answer correctness.

1 INTRODUCTION

Following the release of Deepseek R1 Guo et al. (2025), post-training Large Language Models (LLMs) using Reinforcement Learning with Verifiable Rewards (RLVR) has gained widespread attention. Several works have expanded on reinforcement learning based post-training (Liu et al., 2025b; Zhang & Zuo, 2025; Yu et al., 2025; Liu et al., 2025a), while others have examined its limitations Samineni et al. (2025). Yue et al. (2025) argued Pass@K accuracy of the RL-trained model falls below that of the base model as K increases and Cui et al. (2025) showed that the performance gains can be predicted using entropy of the base model.

However, these works focus on final-answer accuracy. The effect of RLVR on the intermediate tokens or the chain of thought (CoT) that precedes the answer, is less understood, and prior work provides conflicting evidence. Wen et al. (2025) proves that under GRPO, the rate of correct CoTs increases and concludes that RLVR implicitly incentivizes correct reasoning. In contrast, Yu et al. (2026) finds that RLVR improves accuracy without raising causal importance or sufficiency of reasoning.

We argue that RLVR does not reinforce correct reasoning but reinforces any CoTs that increases the probability of a correct final answer. To identify this, we characterize the optimal policy of the KL-regularised objective underlying GRPO (Lemma 1). Using the closed-form expression of the optimal policy we derive the Prompt Augmentation objective Kambhampati et al. (2025), which states that intermediate tokens are an augmentation of the prompt whose only role is to raise the solve

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.

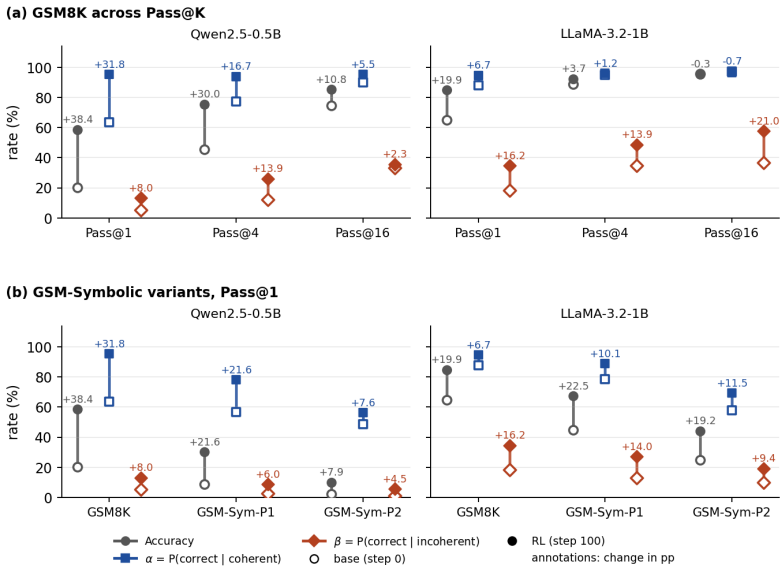

Figure 1: Base (hollow) to RL (filled) for accuracy, $\alpha = P(\text{correct} \mid \text{coherent})$ and $\beta = P(\text{correct} \mid \text{incoherent})$; annotations in percentage points. (a) GSM8K across Pass@K. (b) GSM8K and its GSM-Symbolic variants at Pass@1. Both α and β rises in almost every setting along with accuracy

probability. We show that the policy reweights a CoT according to how much the CoT increases the probability of success relative to the prompt alone (Theorem 1). The update is indifferent to the validity, coherence or faithfulness of the CoT. For any given trace property, the probability of a correct answer can increase from both the CoTs that have the property and for CoTs that lack it. Wen et al. (2025) measure only the first channel and Yu et al. (2026) observe the second.

We derive two testable predictions from this account that the claim of Wen et al. (2025) denies (Section 4). In a formally verifiable domain, RLVR should raise accuracy without raising trace validity (P1). The GRPO results on maze navigation, setting with verifiable traces (Valmeekam et al., 2025) show the increase in accuracy even when no valid traces are available, which is acheived by training the model on swapped traces. In mathematical reasoning, where validity cannot be verified, the rate at which incorrect traces reach the correct answer should rise (P2). To test P2, we study the effect of RLVR on traces using GSM8K (Cobbe et al., 2021). Since formal verification of intermediate reasoning steps is not tractable at scale, we cannot directly evaluate trace validity. Instead, we introduce a proxy metric called **trace coherence**, which reflects the consistency of reasoning steps: we measure it by identifying the errors in the reasoning steps using LLM-as-a-judge, where errors are defined using a First-Order Logic (FOL) framework (Section 4). While trace validity implies coherence, the converse need not hold. Over a set of traces, we report $\alpha = P(\text{correct answer} \mid \text{coherent trace})$ and $\beta = P(\text{correct answer} \mid \text{incoherent trace})$, the latter the answer-shortcut rate, and compare both before and after RLVR training. We define Pass@K trace coherence and evaluate α and β on GSM8K for $K \in \{1, 4, 16\}$, and conduct out-of-distribution evaluation on the GSM8K-Symbolic-P1 and P2 variants. Our results (Section 6) show that β increases in every setting, so P2 holds: RLVR reinforces both the coherent-trace channel and the answer-shortcut channel.

Section 3 derives RLVR as prompt augmentation conditioned on success, Section 4 states the predictions and the evaluation method, and Section 6 reports the results.

2 Related Work

Reasoning traces has been studied since the release of DeepSeek R1 Guo et al. (2025) to better interpret LLMs and their improved task performance. Marjanović et al. (2025) analyse the length, structure, and content of traces generated by DeepSeek R1, for interpret-ability and safety. Further,

the correlation between a trace and the final answer has been tested directly in domains with formal verifiers. In such domains, Valmeekam et al. (2025); Bhambri et al. (2025b) demonstrate a lack of correlation between trace validity and final answer correctness during Supervised Fine-Tuning (SFT), and Valmeekam et al. (2025) further show that GRPO post-training raises plan accuracy without raising trace validity. Arcuschin et al. (2025) find that the trace often does not describe how the answer was reached. Bhambri et al. (2025a) find that making traces more interpretable to people can lower accuracy. Based on these findings, Kambhampati et al. (2025) argue that intermediate tokens should not be anthropomorphized as reasoning, and propose the prompt-augmentation view, that intermediate tokens help the model by raising solve probability, and the model's role is to learn the Skolem function producing such augmentations.

Under RLVR, the evidence on the effect of outcome reward on trace quality is divided. Wen et al. (2025) studied the Pass@K trace correctness for RLVR trained Model and show that the expected GRPO advantage is positive for correct CoTs, so the correct-CoTs rate increases. But they do not measure the rate at which the incorrect-CoTs lead to the correct answer for both the RL model and the base model. While, Yu et al. (2026) introduce metrics for causal importance and sufficiency, which measure whether the final answer depends on the generated reasoning and whether the reasoning alone supports an unambiguous answer and find that RLVR improves accuracy without improving either.

To resolve this disagreement, we look at the outcome-reward objective that RLVR optimises. Under the control-as-inference view of KL-regularised RL, the optimal policy is the base policy reweighted by the likelihood of success (Levine, 2018). Our Theorem 1 shows that it yields the prompt-augmentation view of Kambhampati et al. (2025): a CoT is reinforced only through its effect on the solve probability, whether or not its reasoning is correct.

To test the above predictions on CoTs, we evaluate reasoning traces in the math reasoning domains similar to (Wen et al., 2025). Evaluating reasoning traces for correctness is difficult and prior works employ LLM-as-a-judgeGu et al. (2024), where the LLM is prompted to classify the reasoning traces as correct or incorrect (Wen et al., 2025). However, the LLM judge do not establish trace correctness/formal validity. We instead prompt the LLM to identify specific errors from a fixed taxonomy expressed in first-order logic and define *trace coherence* as the absence of detected errors instead of trace correctness. We also validate the judge against human annotations (Section 4).

3 Theory: RLVR as Learning Prompt Augmentation Conditioned on Success

Setup We consider the RL post training of LLMs, where the auto regressive generation of next token is modeled as a LLM MDP that concatenates the next token to the context leading to deterministic state transition dynamics (Samineni et al., 2025). Further, RLVR considers, the outcome reward, where the model receives a binary reward only at the end of the response and it is split equally across the tokens in the response.

Fix a prompt x. A response is $y = (c, \mathrm{ans})$, where c contains the intermediate tokens, or chain of thought (CoT), and ans is the final answer.

Deterministic State Transitions & Terminal Rewards

1. Consider the current state s_t as the prefix, an action a_t is the next token, and the transition is deterministic: $s_{t+1} = (s_t, a_t)$.
2. The verifier assigns a terminal reward $R(y) \in \{0, 1\}$,

$$R(y) = \begin{cases} 1, & \text{if the answer is correct,} \\ 0, & \text{otherwise.} \end{cases}$$

3. All intermediate rewards are zero. $r(s_t, a_t) = 0$
4. The discount factor, $\gamma = 1$. Since, GRPO assigns uniform credit to every token, the return $\gamma^{T-t} R(y)$ would be the same at every token t, which holds only if $\gamma = 1$.

Then GRPO (Shao et al., 2024) samples $y_1, \ldots, y_G \sim \pi_{\text{old}}(\cdot \mid x)$, where π_{old} is the policy before the current update ($\pi_{\text{old}} = \pi_{\text{base}}$ at the first step). It assigns every token in response y_i the same group-normalised advantage, $A_i = (R_i - \bar{R})/\sigma_R$, and maximises the clipped surrogate

$$J(\theta) = \mathbb{E}\left[\frac{1}{G}\sum_i \min\left(\rho_i A_i,\ \text{clip}(\rho_i, 1-\epsilon, 1+\epsilon)A_i\right)\right], \qquad \rho_i = \frac{\pi_\theta(y_i \mid x)}{\pi_{\text{old}}(y_i \mid x)}. \tag{1}$$

Thus, GRPO assigns uniform credit to all tokens in a sampled response. The question is how this uniform trajectory-level update optimises the reasoning in the CoTs?

Let Sol denote the the event that a response sampled from π_{base} starting from s_t has the final answer correct. For a prefix s_t, define the base policy solve probability,

$$v(s_t) = \mathbb{P}_{\pi_{\text{base}}}(R(y) = 1 \mid s_t) = \mathbb{P}_{\pi_{\text{base}}}(\text{Sol} \mid s_t), \tag{2}$$

Consider the soft value function of a state $V(s) = \log \mathbb{P}_{\pi_{\text{base}}}(\text{Sol} \mid s)$ and soft value of state action value $Q(s,a) = \log \mathbb{P}_{\pi_{\text{base}}}(\text{Sol} \mid s, a)$ for action a at state s.

$$Q(s_t, a_t) = r(s_t, a_t) + \gamma \sum_{s_{t+1}} p(s_{t+1} \mid s_t, a_t)\, V(s_{t+1}) \tag{3}$$

Because the transition is deterministic, $\gamma = 1$ and every intermediate reward is zero. Hence, from the MDP,

$$Q(s_t, a_t) = V(s_{t+1}). \tag{4}$$

Trust Region Objective Let π_{base} be the the base policy and consider the Trust region objective, which raises the expected reward while constraining the KL divergence of the updated policy from previous policy with weight $1/\eta$ (Schulman et al., 2015):

$$\pi_{\text{new}} = \arg\max_\pi \left\{\mathbb{E}_{y\sim\pi(\cdot|x)}[R(y)] - \tfrac{1}{\eta} D_{\text{KL}}\big(\pi(\cdot \mid x) \,\|\, \pi_{\text{base}}(\cdot \mid x)\big)\right\}, \tag{5}$$

The clipped surrogate in GRPO equation 1 is an approximation to keeping the updated policy close to the policy at the previous step. (Schulman et al., 2017).

Lemma 1 (Optimal policy: Base policy conditioned on success.)**.** *The maximiser of equation 5 is* $\pi_{\text{base}}(y \mid x)\, e^{\eta R(y)}/Z$*, and as* $\eta \to \infty$ *it converges to the base policy conditioned on success,*

$$\pi^\star(y \mid x) = \pi_{\text{base}}(y \mid x, \text{Sol}) = \frac{\pi_{\text{base}}(y \mid x)\, \mathbb{1}\{\text{Sol}(y) = 1\}}{\mathbb{P}_{\pi_{\text{base}}}(\text{Sol} \mid x)}, \tag{6}$$

The equation 6 is the optimal policy of control as inference (Levine, 2018), the prior conditioned on the success. It is the base policy scaled by the soft advantage $Q_{\text{soft}} - V_{\text{soft}}$, and for outcome only rewards the soft values coincide with the Q and V, defined above.

The proof is restated in Appendix A.1 for completeness.

We derive the following lemma using equation 4 in equation 6.

Lemma 2 (Optimal policy at Token-level)**.** *For any prefix* s_t *and any token* a*, with* $s_{t+1} = (s_t, a_t)$*,*

$$\log \frac{\pi^\star(a \mid s_t)}{\pi_{\text{base}}(a \mid s_t)} = V(s_{t+1}) - V(s_t). \tag{7}$$

(Proof in Appendix A.2.)

The identity holds for every token in the vocabulary at every prefix and a token is up-weighted when a prefix concatenated with the token has higher solve probability than the prefix without it.

The following theorem extends Eq.equation 7 along the Intermediate Tokens or CoTs, where the per-token differences telescope and difference in the value of CoT and the value of the prompt remains.

Theorem 1 (Outcome-only RLVR as learning prompt augmentation)**.** *For a CoT* $c = (a_0, \ldots, a_{k-1})$ *with* $s_0 = x$*,*

$$\log \frac{\pi^\star(c \mid x)}{\pi_{\text{base}}(c \mid x)} = \sum_{t=0}^{k-1} \big[V(s_{t+1}) - V(s_t)\big] = V(x, c) - V(x) = \log \frac{\mathbb{P}_{\pi_{\text{base}}}(\text{Sol} \mid x, c)}{\mathbb{P}_{\pi_{\text{base}}}(\text{Sol} \mid x)}. \tag{8}$$

Consequently, any property of the intermediate tokens, including validity, coherence, faithfulness have the effect only through $\mathbb{P}(\text{Sol} \mid x, c)$*, i.e final answer correctness given the CoT,* c *and prompt,* x*.*

(Proof in Appendix A.3.)

Corollary 1 (Skolem Function)**.** *The right-hand side of equation 8 is positive when* $\mathbb{P}(\text{Sol} \mid x{+}c) > \mathbb{P}(\text{Sol} \mid x)$*, which is the prompt-augmentation inequality of Kambhampati et al. (2025): intermediate tokens help by raising the solve probability, and the policy that produces such* c *is the Skolem function. Theorem 1 shows that outcome-only RLVR objective learns such a function. It is indifferent to* why *an augmentation works: a CoT is favoured whenever it raises the probability of a correct answer, whether that comes from valid reasoning or from any other scaffold.*

Remark 1 (From $\pi^\star$ to the RL model)**.** $\pi^\star$ *is the optimal policy of the objective equation 5 and* π_{RL} *is the policy GRPO reaches by optimising its clipped surrogate equation 1, and approximates* $\pi^\star$*. Since the objective is indifferent to every property of the CoT beyond its effect on the solve probability (Theorem 1),the objective would be the same and so the theorem applies to the surrogate that approximates it. Whether it does is an empirical question, and Section 4 tests it by comparing the base policy with* π_{RL}*: a CoT is reinforced when it raises the base model's solve probability, indifferent to any property of CoT.*

Summary. Outcome-only RLVR learns prompt augmentation conditioned on success: for each prompt, it reweights the CoTs the base model already produces by how much each raises the probability of a correct answer (Theorem 1), and is indifferent to whether that rise comes from valid reasoning or any other scaffold (Corollary 1). A model trained with GRPO approximates this objective and therefore do not favor valid reasoning (Remark 1). Section 4 turns this into predictions that separate it from the claim that RLVR implicitly incentivises correct reasoning.

4 Implications and Empirical Validation

Wen et al. (2025) study GRPO on mathematical reasoning and conclude that RLVR *implicitly incentivises correct reasoning*. Their argument has a theoretical and an empirical part. Theoretically, they let p_c be the rate of correct CoTs, α the probability of a correct answer given a correct CoT and β the probability given an incorrect one, and show that when $\alpha > \beta$, the expected advantage of the correct CoTs is positive and prove that GRPO raises the its rate, p_c. They further argue that as training proceeds α rises while β falls, as spurious correlations are trained away. Empirically, they prompt an LLM judge to label each CoT correct or incorrect and report CoT-Pass@K, which is an entangled measure of CoT correctness computed on only traces with correct final answer and equals $\alpha\, p_c$ at $K = 1$. So a rise in it cannot be interpreted as RL increases correct CoTs alone. They also report $P(\text{CC} \mid \text{CA})$, the fraction of correct answers whose CoT is correct. Both metrics count a trace only when its answer is correct or when CoT is correct, but incorrect CoTs that end in correct answer or incorrect answer which measure β, are not reported. The conjecture that β falls is therefore untested by their own metrics.

4.1 Implications of Theorem 1 for CoT Reasoning

Let F be any property of the CoT and define $\alpha = \mathbb{P}(\text{Sol} \mid x, F)$ and $\beta = \mathbb{P}(\text{Sol} \mid x, \neg F)$. Wen et al. (2025) show that when $\alpha > \beta$ the rate of F rises, since responses with F then carry positive expected advantage. But, our Theorem 1 says that same rule applies to every CoT: one *without* F that ends in a correct answer receives the same weight as one with F for the same solve probability,

$v(c)$. CoTs that lack the property but reach the answer are therefore reinforced alongside those that have it. The objective does not reinforce F alone. Their argument uses no fact about F other than $\alpha > \beta$, so it holds equally for properties unrelated to reasoning; a rise in p_c cannot by itself show that correct reasoning is rewarded.

4.2 Testable Predictions

Theorem 1 and Lemma 1 give a prediction that the claim of Wen et al. (2025) denies, and we test it in two kinds of domain.

- **P1.** *Where trace validity can be verified.* Incorrect traces that reach the answer are up-weighted like correct ones, so β rises. If the base model produces no valid traces, RLVR raises accuracy while validity stays at zero.
- **P2.** *Where trace validity cannot be verified.* The same holds for any answer-independent measure of the trace, here we define coherence of the trace as trace validity cannot be established: incoherent traces that reach the answer are reinforced, so β rises.

If RLVR rewarded only reasoning and not as a Prompt augmentation conditioned only on success, β would not increase, as Wen et al. (2025) argue.

4.3 P1: Evidence from a Formally Verifiable Domain

Valmeekam et al. (2025) train transformers from scratch with SFT on maze-solving traces, where trace validity can be checked, and post-train them with GRPO (their Section 6.1). Plan accuracy rises without a rise in trace validity. For models trained on Searchformer-style traces, validity falls from 89.0% to 52.1% on Wilson mazes while plan accuracy rises from 40.8% to 63.7% (their Table 5). Models trained on traces swapped across problems have 0% valid traces at every checkpoint, yet GRPO raises their plan accuracy from 83.3% to 99.5% (their Table 4). This contradicts the claim that RLVR incentivises correct reasoning and P1 holds.

4.4 P2: Test in Mathematical Reasoning

Formal verifiers for trace validity do not exist for math-word problems, where the claim that RLVR incentivises reasoning is made. So, we use the similar setting as (Wen et al., 2025), a math domain where an LLM as judge is used for evaluating the traces. Analyzing math word problems presents challenges in defining trace validity. To address this, we adopt an error taxonomy and introduce a metric of Trace Coherence, the absence of errors, inplace of Trace Correctness. We define Pass@K Trace coherence independent of answer correctness, so that the metrics: p_c, α, and β can be measured. Below we describe the error taxonomy, the metrics, and the trace evaluation using LLM-as-Judge.

4.5 Error Categories for Mathematical Reasoning

We consider GSM8K problems, that take two to eight steps of elementary calculations. Each consists of given facts, a formula for the answer and intermediate quantities, a structure that maps with the construction of First-Order Logic (FOL). Using the FOL based error taxonomy, we define the error types in Table 1.

Table 1: Error categories and their subtypes.

Error Category	**Description / Subtypes**
False Premise	Conceptual Misunderstanding; Semantic Error; False Assumption; Units Misinterpretation; Incorrect Derivation Step from problem description;
False Rule	Type / Operand Mismatch; Inference Violation; Operation Misapplication; Quantifier Misuse; Missing Necessary Steps
Calculator Error	Simple numeric or arithmetic mistakes

Table 2: GPT-4o Evaluation Metrics on Human Annotated Dataset (accuracy 81.1%, $\kappa = 0.62$)

Class	Recall	Precision	F1	Support
Incoherent Trace (Presence of Error)	0.84	0.75	0.79	75
Coherent Trace (No error)	0.79	0.87	0.83	100

4.6 Trace Evaluation with LLM as Judge.

We use LLMs-as-judge for tagging the errors in the traces collected from RL post training. Direct classification of errors from raw traces may not be reliable. We leverage the translative capabilities of LLMs, particularly prompting GPT-4o to convert each response into a First Order Logic representation and classify errors according to our taxonomy provided in Table 1 (see Appendix E for details).

4.7 Validation Accuracy of LLMs for Trace Evaluation

LLM for Evaluation: We employ GPT-4o to classify errors in the reasoning traces. The prompts used for evaluation are provided in Appendix D.

Evaluation Dataset: To assess the LLM's classification accuracy for error tagging, we curated a human-annotated dataset. It contains 25 responses for each error type and 100 responses with no errors. Each response with an error tag is annotated with one error type.

Evaluation Accuracy: Table 2 shows the LLM's evaluation accuracy on human-annotated responses curated as described in experimental setup. The accuracy reported is 81.1%.

4.8 Trace Coherence and Pass@K

We define Pass@K Trace coherence as follows

Pass@K Trace Coherence: For each problem, we consider the set of k sampled reasoning traces. The set is counted as coherent if at least one trace is error-free, and incoherent if all k traces contain at least one error.

We also define p_c, α, and β using Trace coherence and study each of them independently during RLVR Post training.

- $p_c = P(\text{coherent trace})$ the rate at which the model produces coherent trace.
- $\alpha = P(\text{Correct Ans} \mid \text{Coherent trace})$ The rate at which the trace has correct final answer given the trace is coherent.
- $\beta = P(\text{Correct Ans} \mid \text{Incoherent trace})$ The rate at which trace has correct final answer given the trace has errors (answer-shortcut rate).

5 Experimental Setup

Datasets: We perform RL training on the *GSM8K* dataset Cobbe et al. (2021), a widely used benchmark of grade-school mathematical reasoning problems. The dataset contains approximately 8.5K problems, each consisting of a question–answer pair, with 7.5K training examples and 1319 test examples. For trace coherence analysis, we evaluate both the base model and the RL-trained model on the GSM8K test set.

To further assess the robustness of reasoning in CoTs under distribution shifts, we additionally analyse the trace coherence of both the base model and RL-trained model on two out-of-distribution variants: *GSM8K-Symbolic-P1* and *GSM8K-Symbolic-P2* by Mirzadeh et al. (2024). These variants are constructed by augmenting each problem with one (P1) or two (P2) additional natural-language clauses. While these clauses increase surface and compositional complexity, they typically do not alter the underlying mathematical solution.

Table 3: GSM8K across Pass@K ($N = 1319$). p_c is the coherence rate, $\alpha = P(\text{correct} \mid \text{coherent})$ and $\beta = P(\text{correct} \mid \text{incoherent})$. Base is the model at Step 0 and RL the model at Step 100. Levels are percentages; Δ is the RL value minus the base value, in percentage points, so a positive Δ means the quantity rose under RLVR. A problem counts as coherent (correct) if at least one of its K traces is coherent (correct). β rises at every K for both models.

	Accuracy			p_c			α			β		
Dataset	Base	RL	ΔAcc	Base	RL	Δp_c	Base	RL	$\Delta\alpha$	Base	RL	$\Delta\beta$
Qwen2.5-0.5B												
GSM8K Pass@1	20.2	58.6	+38.4	25.8	55.3	+29.5	63.5	95.3	**+31.8**	5.2	13.2	+8.0
GSM8K Pass@4	45.4	75.4	+30.0	51.2	72.8	+21.6	77.3	94.0	**+16.7**	12.1	26.0	+13.9
GSM8K Pass@16	74.5	85.3	+10.8	72.7	83.2	+10.5	89.8	95.3	**+5.5**	33.4	35.7	+2.3
LLaMA-3.2-1B												
GSM8K Pass@1	64.9	84.8	+19.9	66.9	83.8	+16.9	87.9	94.6	+6.7	18.4	34.6	**+16.2**
GSM8K Pass@4	88.7	92.4	+3.7	89.7	92.3	+2.6	94.9	96.1	+1.2	34.8	48.7	**+13.9**
GSM8K Pass@16	95.6	95.3	−0.3	97.2	96.6	−0.6	97.3	96.6	−0.7	36.7	57.7	**+21.0**

Table 4: GSM8K-Symbolic variants at Pass@1 ($N = 1000$; 998 for LLaMA on P1). GSM8K at Pass@1 is in Table 3. p_c is the coherence rate, $\alpha = P(\text{correct} \mid \text{coherent})$ and $\beta = P(\text{correct} \mid \text{incoherent})$. Base is the model at Step 0 and RL the model at Step 100. Levels are percentages; Δ is the RL value minus the base value, in percentage points, so a positive Δ means the quantity rose under RLVR. Under distribution shift β rises in every setting.

	Accuracy			p_c			α			β		
Dataset	Base	RL	ΔAcc	Base	RL	Δp_c	Base	RL	$\Delta\alpha$	Base	RL	$\Delta\beta$
Qwen2.5-0.5B												
GSM8K-Symbolic-P1	8.6	30.2	+21.6	11.0	31.0	+20.0	56.5	78.1	**+21.6**	2.7	8.7	+6.0
GSM8K-Symbolic-P2	2.1	10.0	+7.9	1.9	8.6	+6.7	48.5	56.1	**+7.6**	1.2	5.7	+4.5
LLaMA-3.2-1B												
GSM8K-Symbolic-P1	44.8	67.3	+22.5	48.3	65.2	+16.9	78.7	88.8	+10.1	13.1	27.1	**+14.0**
GSM8K-Symbolic-P2	25.0	44.2	+19.2	31.7	50.0	+18.3	57.7	69.2	**+11.5**	9.8	19.2	+9.4

Training: Our experiments use Qwen-2.5-0.5B from the Qwen-2.5 family and LLaMa-3.2-1B as the base models. The models are fine-tuned using the VERL pipeline Sheng et al. (2025) with GRPO and evaluated at steps 0 and 100. All hyperparameters are listed in Appendix B

6 Results and Discussion

This section tests P2 (Section 4.2): if RLVR learns prompt augmentation conditioned on success, β rises; if it reinforces the correct reasoning, β falls, as Wen et al. (2025) argue.

We measure the coherence rate p_c, α, and β for the base model and for the RL model after RLVR post-training, for Qwen2.5-0.5B and LLaMA-3.2-1B. Table 3, reports these rates along with accuracy on the GSM8K test set for different Pass@K values, namely $K \in \{1, 4, 16\}$. Table 4 reports them on two OOD datasets: GSM8K-Symbolic-P1, which adds one extra clause per problem, and GSM8K-Symbolic-P2, which adds two extra clauses per problem. and we evaluated for $K = 1$. Additional metrics also provided in Appendix C.

Figures 1(a) and 1(b) plot accuracy, the change of α and β from base model to the RL model for both the settings: across Pass@K and on the OOD variants. The figures show the finding of this section that RLVR raises not only accuracy and α but also β, in every setting, for both the models.

6.1 In-distribution GSM8K: Trace Coherence across pass@k

Table 3 and Figure 1(a) report the results for different Pass@K. At $K = 1$ each problem has one trace, so α and β are the rates at which a coherent and an incoherent trace, respectively, produce the correct answer. At $K > 1$ a problem counts as coherent if any of its K traces is coherent and as

correct if any is correct, and these can be different traces, so coherence becomes a property of the set rather than of a single trace.

In all settings of Pass@K, the RL model shows an increase in accuracy, in p_c, and in the rate of correct answers following both coherent and incoherent traces, the one exception is LLaMA-3.2-1B at $K = 16$, where accuracy is already at its ceiling (95%).

On Qwen2.5-0.5B, correct answers from both coherent and incoherent traces rise relative to the base model at every K. The increase in α falls from $+31.8$ pp at $K = 1$ to $+5.5$ pp at $K = 16$, while the largest increase in β, $+13.9$ pp, is at $K = 4$.

On LLaMA-3.2-1B, the increase in β exceeds the increase in α at every K: Which means that the increase in the empirical rate of correct answers from incoherent traces is therefore greater than the increase from coherent traces.

6.2 Trace Coherence on Out of Distribution: GSM-Symbolic

Table 4 and Figure 1(b) report the results for the GSM8K-Symbolic variants. On both variants, the RL model shows an increase in accuracy, in p_c, the fraction of coherent traces, and in the rate of correct answers following both coherent and incoherent traces. β rises by 6.0 and 4.5 percentage points for Qwen2.5-0.5B on P1 and P2, and by 14.0 and 9.4 points for LLaMA-3.2-1B. The OOD results also show an increase in contribution of incoherent traces to final-answer correctness.

These observations show that RLVR does not implicitly incentivize reasoning. GRPO improves final-answer correctness, and to do so it can raise the rate of coherent traces, the rate at which incoherent traces reach a correct answer, or both. In our measurements it raises both channels in all settings in GSM8K and its variants. Therefore P2 holds, and together with P1 (Section 4.3) it contradicts the claim that RLVR implicitly incentivises correct reasoning.

7 Conclusion

Our theoretical formulation and empirical results show that RLVR can increase final-answer accuracy through both correct and incorrect reasoning traces. Prior work presents conflicting views on whether RLVR improves reasoning or mainly exploits answer shortcuts; our results show that both channels coexist. This reframes RLVR as learning prompt augmentation conditioned on final-answer correctness, where intermediate tokens are treated as scaffolding for the model to reach the correct answer, regardless of whether the trace itself is correct. Thus a CoT produced by RL Model is best read as an augmentation that raises the probability of a correct answer, not as an account of how that answer was reached.

AI use statement

In this work, we used generative AI tools for for writing assistance: editing and restructuring the text of the methodology and results sections.

Additionally, we used generative AI tools for evaluating the the traces for errors using GPT-4o acting as a judge, as described in Section 4; this is part of the method, not an aid to writing it, and its agreement with human annotation is reported in Table 2.

We defined the research question, derived the theoretical results, setup the experimental design and the human annotation of the judge-validation.

We reviewed all AI-assisted text and code: every reported number was recomputed from the source matrices, and we checked the analysis code. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

Ethics statement

This work analyses the reasoning traces of openly available language models on public mathematics benchmarks (GSM8K and GSM8K-Symbolic). It involves no human subjects and no personal or sensitive data.

Reproducibility statement

The theoretical results are stated in Section 3 with complete proofs in Appendix A. The models (Qwen2.5-0.5B, LLaMA-3.2-1B), datasets (GSM8K, GSM8K-Symbolic-P1/P2) and training framework (VERL) are publicly available; all training and evaluation hyperparameters are listed in Section 5. The full judge prompt is given in Appendix B. and the error taxonomy in Appendix E. Every number in Section 6 is computed from the coherence matrices by the scripts that would be released with the paper.

References

Iván Arcuschin, Jett Janiak, Robert Krzyzanowski, Senthooran Rajamanoharan, Neel Nanda, and Arthur Conmy. Chain-of-thought reasoning in the wild is not always faithful. *arXiv preprint arXiv:2503.08679*, 2025.

Siddhant Bhambri, Upasana Biswas, and Subbarao Kambhampati. Do cognitively interpretable reasoning traces improve llm performance? *arXiv preprint arXiv:2508.16695*, 2025a.

Siddhant Bhambri, Upasana Biswas, and Subbarao Kambhampati. Interpretable traces, unexpected outcomes: Investigating the disconnect in trace-based knowledge distillation. *arXiv preprint arXiv:2505.13792*, 2025b.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Ganqu Cui, Yuchen Zhang, Jiacheng Chen, Lifan Yuan, Zhi Wang, Yuxin Zuo, Haozhan Li, Yuchen Fan, Huayu Chen, Weize Chen, et al. The entropy mechanism of reinforcement learning for reasoning language models. *arXiv preprint arXiv:2505.22617*, 2025.

Jiawei Gu, Xuhui Jiang, Zhichao Shi, Hexiang Tan, Xuehao Zhai, Chengjin Xu, Wei Li, Yinghan Shen, Shengjie Ma, Honghao Liu, et al. A survey on llm-as-a-judge. *arXiv preprint arXiv:2411.15594*, 2024.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Subbarao Kambhampati, Karthik Valmeekam, Siddhant Bhambri, Vardhan Palod, Lucas Saldyt, Kaya Stechly, Soumya Rani Samineni, Durgesh Kalwar, and Upasana Biswas. Position: Stop anthropomorphizing intermediate tokens as reasoning/thinking traces! *arXiv preprint arXiv:2504.09762*, 2025.

Sergey Levine. Reinforcement learning and control as probabilistic inference: Tutorial and review. *arXiv preprint arXiv:1805.00909*, 2018.

Mingjie Liu, Shizhe Diao, Ximing Lu, Jian Hu, Xin Dong, Yejin Choi, Jan Kautz, and Yi Dong. Prorl: Prolonged reinforcement learning expands reasoning boundaries in large language models. *arXiv preprint arXiv:2505.24864*, 2025a.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. Understanding r1-zero-like training: A critical perspective. *arXiv preprint arXiv:2503.20783*, 2025b.

Sara Vera Marjanović, Arkil Patel, Vaibhav Adlakha, Milad Aghajohari, Parishad BehnamGhader, Mehar Bhatia, Aditi Khandelwal, Austin Kraft, Benno Krojer, Xing Han Lù, et al. Deepseek-r1 thoughtology: Let's think about llm reasoning. *arXiv preprint arXiv:2504.07128*, 2025.

Iman Mirzadeh, Keivan Alizadeh, Hooman Shahrokhi, Oncel Tuzel, Samy Bengio, and Mehrdad Farajtabar. Gsm-symbolic: Understanding the limitations of mathematical reasoning in large language models. *arXiv preprint arXiv:2410.05229*, 2024.

Soumya Rani Samineni, Durgesh Kalwar, Karthik Valmeekam, Kaya Stechly, and Subbarao Kambhampati. Rl in name only? analyzing the structural assumptions in rl post-training for llms. *arXiv preprint arXiv:2505.13697*, 2025.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International conference on machine learning*, pp. 1889–1897. Pmlr, 2015.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. Hybridflow: A flexible and efficient rlhf framework. In *Proceedings of the Twentieth European Conference on Computer Systems*, pp. 1279–1297, 2025.

Manan Tomar, Lior Shani, Yonathan Efroni, and Mohammad Ghavamzadeh. Mirror descent policy optimization. *arXiv preprint arXiv:2005.09814*, 2020.

Karthik Valmeekam, Vardhan Palod, Kaya Stechly, Atharva Gundawar, and Subbarao Kambhampati. Beyond semantics: The unreasonable effectiveness of reasonless intermediate tokens. *arXiv preprint arXiv:2505.13775*, 2025.

Xumeng Wen, Zihan Liu, Shun Zheng, Shengyu Ye, Zhirong Wu, Yang Wang, Zhijian Xu, Xiao Liang, Junjie Li, Ziming Miao, et al. Reinforcement learning with verifiable rewards implicitly incentivizes correct reasoning in base llms. *arXiv preprint arXiv:2506.14245*, 2025.

Qinan Yu, Alexa Tartaglini, Peter Hase, Carlos Guestrin, and Christopher Potts. Outcome rewards do not guarantee verifiable or causally important reasoning. *arXiv preprint arXiv:2604.22074*, 2026.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, et al. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025.

Yang Yue, Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Shiji Song, and Gao Huang. Does reinforcement learning really incentivize reasoning capacity in llms beyond the base model? *arXiv preprint arXiv:2504.13837*, 2025.

Jixiao Zhang and Chunsheng Zuo. Grpo-lead: A difficulty-aware reinforcement learning approach for concise mathematical reasoning in language models. *arXiv preprint arXiv:2504.09696*, 2025.

A Proofs for Section 3

A.1 Proof of Lemma 1

(i) Closed form. Let $\tilde{\pi}(y) = \pi_{\text{base}}(y \mid x) e^{\eta R(y)}/Z$. For any $\pi \ll \pi_{\text{base}}$,

$$\mathbb{E}_\pi[R] - \tfrac{1}{\eta} D_{\text{KL}}(\pi \| \pi_{\text{base}}) = -\tfrac{1}{\eta}\, \mathbb{E}_\pi\Big[\log \frac{\pi(y)}{\pi_{\text{base}}(y \mid x)\, e^{\eta R(y)}} \Big] = -\tfrac{1}{\eta}\, D_{\text{KL}}(\pi \| \tilde{\pi}) + \tfrac{1}{\eta} \log Z,$$

and the second term does not depend on π, so the objective is maximised uniquely at $\pi = \tilde{\pi}$. This is the standard KL-regularised solution (Tomar et al., 2020). *(ii) Limit.* Since $R(y) \in \{0, 1\}$, $\tilde{\pi}(y) = \pi_{\text{base}}(y \mid x)\, e^{\eta \text{Sol}(y)} \big/ \big(p\, e^{\eta} + (1-p)\big)$ with $p = \mathbb{P}_{\pi_{\text{base}}}(\text{Sol} \mid x)$. As $\eta \to \infty$ the mass on $\text{Sol}(y) = 0$ vanishes and $\tilde{\pi}(y) \to \pi_{\text{base}}(y \mid x)\text{Sol}(y)/p = \pi_{\text{base}}(y \mid x, \text{Sol})$, which is equation 6. □

A.2 Proof of Lemma 2

For a prefix s write $\pi(s \mid x) = \sum_{y \succeq s} \pi(y \mid x)$ for the probability that a response begins with s; the next-token conditional of any policy is then $\pi(a \mid s) = \pi(sa \mid x)/\pi(s \mid x)$.

Step 1 (prefix probability under $\pi^\star$). Summing equation 6 over responses that begin with s,

$$\pi^\star(s \mid x) = \sum_{y \succeq s} \frac{\pi_{\text{base}}(y \mid x)\,\text{Sol}(y)}{\mathbb{P}_{\pi_{\text{base}}}(\text{Sol} \mid x)} = \frac{\pi_{\text{base}}(s \mid x)}{\mathbb{P}_{\pi_{\text{base}}}(\text{Sol} \mid x)} \sum_{y \succeq s} \pi_{\text{base}}(y \mid x, s)\,\text{Sol}(y) = \frac{\pi_{\text{base}}(s \mid x)\,\mathbb{P}_{\pi_{\text{base}}}(\text{Sol} \mid s)}{\mathbb{P}_{\pi_{\text{base}}}(\text{Sol} \mid x)}. \tag{9}$$

Step 2 (next-token conditional). Apply equation 9 to s_t and to $s_{t+1} = s_t a$ and divide; the common factor $\mathbb{P}_{\pi_{\text{base}}}(\text{Sol} \mid x)$ cancels:

$$\pi^\star(a \mid s_t) = \frac{\pi^\star(s_{t+1} \mid x)}{\pi^\star(s_t \mid x)} = \frac{\pi_{\text{base}}(s_{t+1} \mid x)}{\pi_{\text{base}}(s_t \mid x)} \cdot \frac{\mathbb{P}_{\pi_{\text{base}}}(\text{Sol} \mid s_{t+1})}{\mathbb{P}_{\pi_{\text{base}}}(\text{Sol} \mid s_t)} = \pi_{\text{base}}(a \mid s_t)\,\frac{\mathbb{P}_{\pi_{\text{base}}}(\text{Sol} \mid s_{t+1})}{\mathbb{P}_{\pi_{\text{base}}}(\text{Sol} \mid s_t)}. \tag{10}$$

Step 3 (take logarithms and use equation 4). Dividing by $\pi_{\text{base}}(a \mid s_t)$ and taking logarithms, $\log\big[\pi^\star(a \mid s_t)/\pi_{\text{base}}(a \mid s_t)\big] = \log \mathbb{P}_{\pi_{\text{base}}}(\text{Sol} \mid s_{t+1}) - \log \mathbb{P}_{\pi_{\text{base}}}(\text{Sol} \mid s_t) = V(s_{t+1}) - V(s_t)$. Equivalently, the numerator is $Q(s_t, a) = \log \mathbb{P}_{\pi_{\text{base}}}(\text{Sol} \mid s_t, a)$, which equals $V(s_{t+1})$ by equation 4. □

A.3 Proof of Theorem 1

Step 1 (chain rule). Both policies are autoregressive, so the probability of the Trace is the product of its next-token probabilities:

$$\pi^\star(c \mid x) = \prod_{t=0}^{k-1} \pi^\star(a_t \mid s_t), \qquad \pi_{\text{base}}(c \mid x) = \prod_{t=0}^{k-1} \pi_{\text{base}}(a_t \mid s_t). \tag{11}$$

Step 2 (ratio of products is product of ratios).

$$\frac{\pi^\star(c \mid x)}{\pi_{\text{base}}(c \mid x)} = \prod_{t=0}^{k-1} \frac{\pi^\star(a_t \mid s_t)}{\pi_{\text{base}}(a_t \mid s_t)}. \tag{12}$$

Step 3 (take logarithms and substitute Lemma 2). The product becomes a sum, and each term is given by equation 7:

$$\log \frac{\pi^\star(c \mid x)}{\pi_{\text{base}}(c \mid x)} = \sum_{t=0}^{k-1} \log \frac{\pi^\star(a_t \mid s_t)}{\pi_{\text{base}}(a_t \mid s_t)} = \sum_{t=0}^{k-1} \big[V(s_{t+1}) - V(s_t)\big]. \tag{13}$$

Step 4 (telescope). Writing the sum out term by term,

$$\begin{aligned} \sum_{t=0}^{k-1} \big[V(s_{t+1}) - V(s_t)\big] &= \big[V(s_1) - V(s_0)\big] + \big[V(s_2) - V(s_1)\big] + \cdots + \big[V(s_k) - V(s_{k-1})\big] \\ &= V(s_k) - V(s_0), \end{aligned} \tag{14}$$

Step 5 (identify the endpoints). The initial state is the prompt, $s_0 = x$, and the final state is the prompt followed by the whole chain, $s_k = (x, c)$. Hence $V(s_k) - V(s_0) = V(x, c) - V(x)$, and by the definition of V this equals $\log \mathbb{P}_{\pi_{\text{base}}}(\text{Sol} \mid x, c) - \log \mathbb{P}_{\pi_{\text{base}}}(\text{Sol} \mid x)$, which is equation 8. The right-hand side is a function of $v(c) = \mathbb{P}_{\pi_{\text{base}}}(\text{Sol} \mid x, c)$ and of the constant $p = \mathbb{P}_{\pi_{\text{base}}}(\text{Sol} \mid x)$ alone; no intermediate state survives, which is the stated consequence. □

B Training and Evaluation Hyperparameters

The training batch size is set to 64, with a mini-batch size of 8. We sample 5 responses per question prompt. During training, the response rollouts for each question are generated with a temperature of

0.6., The maximum prompt length is set to 512 for GSM8K, while the maximum response length is fixed at 1024. The learning rate is set to 1e-6. And, we set the KL divergence coefficient to $1e-3$. All experiments on the GSM8K dataset are conducted using a single A100 80GB GPU. Both the Qwen and Llama family models are trained for 100 global time steps, corresponding to 4 epochs. Model evaluation is performed at two time steps (0 and 100) on the GSM8K test dataset and GSM-Symbolic dataset. For response sampling during evaluation, we set the temperature to 1.0 and top-p to 0.95.

C Additional Metrics

Table 5: Accuracy split by the trace that produced the answer, GSM8K across Pass@K ($N = 1319$). Base is the model at Step 0 and RL the model at Step 100. Levels are percentages; Δ is the RL value minus the base value, in percentage points, so a positive Δ means the quantity rose under RLVR. Accuracy equals, exactly, the fraction of problems solved via a coherent trace ($p_c\alpha$, the CoT-Pass@1 of Wen et al. (2025)) plus the fraction solved via an incoherent trace ($(1-p_c)\beta$). A problem counts as coherent (correct) if at least one of its K traces is coherent (correct).

	Accuracy			Solved via a trace that is: coherent $p_c\alpha$			Solved via a trace that is: incoherent $(1-p_c)\beta$		
GSM8K	Base	RL	ΔAcc	Base	RL	$\Delta(p_c\alpha)$	Base	RL	$\Delta((1-p_c)\beta)$
Qwen2.5-0.5B									
Pass@1	20.2	58.6	+38.4	16.4	52.7	+36.3	3.9	5.9	+2.0
Pass@4	45.4	75.4	+30.0	39.5	68.3	+28.8	5.9	7.1	+1.2
Pass@16	74.5	85.3	+10.8	65.4	79.3	+13.9	9.1	6.0	−3.1
LLaMA-3.2-1B									
Pass@1	64.9	84.8	+19.9	58.8	79.2	+20.4	6.1	5.6	−0.5
Pass@4	88.7	92.4	+3.7	85.1	88.7	+3.6	3.6	3.7	+0.2
Pass@16	95.6	95.3	−0.3	94.6	93.3	−1.3	1.0	2.0	+1.0

Table 6: Accuracy split by the trace that produced the answer, GSM8K-Symbolic variants at Pass@1 ($N = 1000$; 998 for LLaMA on P1). GSM8K at Pass@1 is in Table 5. Base is the model at Step 0 and RL the model at Step 100. Levels are percentages; Δ is the RL value minus the base value, in percentage points, so a positive Δ means the quantity rose under RLVR. Accuracy equals, exactly, the fraction of problems solved via a coherent trace ($p_c\alpha$, the CoT-Pass@1 of Wen et al. (2025)) plus the fraction solved via an incoherent trace ($(1-p_c)\beta$).

	Accuracy			Solved via a trace that is: coherent $p_c\alpha$			Solved via a trace that is: incoherent $(1-p_c)\beta$		
Dataset	Base	RL	ΔAcc	Base	RL	$\Delta(p_c\alpha)$	Base	RL	$\Delta((1-p_c)\beta)$
Qwen2.5-0.5B									
GSM8K-Symbolic-P1	8.6	30.2	+21.6	6.2	24.2	+18.0	2.4	6.0	+3.6
GSM8K-Symbolic-P2	2.1	10.0	+7.9	0.9	4.8	+3.9	1.2	5.2	+4.0
LLaMA-3.2-1B									
GSM8K-Symbolic-P1	44.8	67.3	+22.5	38.0	57.9	+19.9	6.8	9.4	+2.6
GSM8K-Symbolic-P2	25.0	44.2	+19.2	18.3	34.6	+16.3	6.7	9.6	+2.9

D PROMPT FOR ERROR CATEGORIZATION FOR GSM8K DATASET

ERROR TAGGING PROMPT

You are an expert mathematical error analyzer. Your task is to identify specific error types in the model's solution to a math word problem.

ANALYSIS PROCESS:

1. Read the problem carefully and understand what is being asked.
2. Verify the model response by converting it into First-Order Logic (FOL) and identifying any errors based on the taxonomy below.
3. You MUST return a valid JSON object with exactly these three keys: `"First-Order Logic"`, `"error_tags"`, `"rationale"`.

ERROR TAXONOMY (use these exact labels):

1. **False Premise** - The model makes incorrect assumptions or misinterprets the problem:
   - Assumes data not given in the problem
   - Misinterprets key terms or relationships
   - Ignores important given information
   - Uses wrong units or constants (e.g., week $\neq$ 7 days)
2. **False Rule** - The model applies incorrect mathematical logic:
   - Uses wrong formulas or operations
   - Applies operations to incompatible types
   - Makes logical inference errors
   - Misunderstands mathematical relationships
3. **Missing Steps** - The model skips necessary reasoning steps:
   - Jumps to conclusion without showing work
   - Omits intermediate calculations needed for the answer
   - Only flag if the step is truly necessary for correctness
4. **Calculator Error** - The model makes wrong addition, subtraction, multiplication, or division
   - ONLY flag if using a caluclator is gone wrong (e.g., $5 \times 6 = 10$ instead of 30)
   - Do NOT flag if calculations are correct but the final answer differs from expected
   - Do NOT flag if the model uses correct math but applies it to wrong numbers
5. **Format Error** - The model's output format is wrong:
   - Final answer not in `\boxed{{{number}}}` format
   - Extra text around the answer
   - Missing or malformed answer box
6. **None** - No errors found (use empty list []):
   - All calculations are correct
   - Proper format used
   - All necessary steps shown
   - Correct interpretation of problem

Examples

1. **Problem:** "In a survey, 40% of students said they like basketball, 30% said they like chess, and 20% said they like both. If 200 students were interviewed, how many students like either basketball or chess?"
   Model Response:

```
Number of students who like basketball = 200 × 0.4 = 80
Number of students who like chess = 200 × 0.3 = 60
Total students who like basketball or chess = 80 + 60 = 140
↪  \boxed{{140}}
```

error_tags: ["False Rule"]

rationale: The model incorrectly assumes the sets of basketball and chess fans are disjoint. Since 20% of students like both sports, the correct total should subtract the overlap: $80 + 60 - 40 = 100$. Adding the numbers directly violates the rules of set union, leading to an overcount.

2. **Problem:** "Today, at the school science lesson, Jake learned that there are 8 more solar systems for every planet in the galaxy. If there are 20 planets in the galaxy, how many solar systems and planets are there altogether?"

   Model Response:

   ```
   The galaxy has 20 planets. Each planet has 8 solar systems.
   Total solar systems = 20 × 9 = 180.
   Total planets + solar systems = 180 + 20 = 200. \boxed{{200}}
   ```

   error_tags: ["False Premise"]

   rationale: The model misinterprets the problem by considering 9 more solar systems for every planet as 9 solar systems per planet instead of 8.

3. **Problem:** "If Alice purchased 3600 acorns for nine times the price Bob paid, and Bob paid $6000 for his acorns, how much did Alice pay for each acorn?"

   Model Response:

   ```
   Total Alice paid = 9 × Total Bob paid = 9 × 6000 = 54000
   Price per acorn for Alice = 15 \boxed{{15}}
   ```

   error_tags: ["Missing Step"]

   rationale: The model correctly computes the total amount Alice paid but skips the division step that shows how the per-acorn price is calculated: Price per acorn for Alice = 54000 / 3600 = 15. This missing step is necessary to make the logical reasoning complete.

4. **Problem:** "Erica sees 9 butterflies in the garden. She sees one-third of them fly away. How many butterflies are left in the garden?"

   Model Response:

   ```
   One-third of 9 = 4. Remaining butterflies = 9 - 4 = 5.
   ↪  \boxed{{5}}
   ```

   error_tags: ["Calculator Error"]

   rationale: The calculation 'one-third of 9 = 4' is mathematically wrong. It should be 9 ÷ 3 = 3.

5. **Problem:** "5 geckos eat 6 insects each. 3 lizards eat twice as much as the geckos. How many total insects were eaten?"

   Model Response:

   ```
   Geckos eat: 5 × 6 = 30 insects.
   Lizards eat: 3 × 12 = 36 insects.
   Total: 30 + 36 = 66 insects. \boxed{{66}}
   ```

   error_tags: []

   rationale: All arithmetic calculations are mathematically correct: 5 × 6 = 30, 3 × 12 = 36, and 30 + 36 = 66. No arithmetic errors present. Given data is used correctly. No False premise, rule applied properly and all steps are used correctly.

6. **Problem:** "Betty is saving money for a new wallet which costs $100. Betty has only half of the money she needs. Her parents decided to give her $15 for that purpose, and her grandparents twice as much as her parents. How much more money does Betty need to buy the wallet?"

   Model Response:

```
In the beginning, Betty has only 100 / 2 = $<<100/2=50>>50.
Betty's grandparents gave her 15 * 2 = $<<15*2=30>>30.
This means, Betty needs 100 - 50 - 30 - 15 =
↪  $<<100-50-30-15=5>>5 more.
 #### 5
```

error_tags: ["Format Error"]

rationale: The model response is not in the correct format $\boxed{5}$.

7. **Problem:** "There are 4 tables with 4 legs each, 1 sofa, 2 chairs with 4 legs each, 3 tables with 3 legs each, 1 table with 1 leg, and 1 rocking chair with 2 legs. How many legs total?"

   Model Response:

   ```
   4 tables × 4 legs = 16, 1 sofa × 4 legs = 4,
   2 chairs × 4 legs = 8, 3 tables × 3 legs = 9,
   1 table × 1 leg = 1, 1 rocking chair × 2 legs = 2.
   Total: 16 + 4 + 8 + 9 + 1 + 2 = 40 legs. \boxed{{40}}
   ```

   error_tags: []

   rationale: The model makes reasonable assumptions (sofa has 4 legs) based on common knowledge. All calculations are correct.

8. **Problem:** "If Alice purchased 3600 acorns for nine times the price Bob paid, and Bob paid $6000 for his acorns, how much did Alice pay for each acorn?"

   Model Response:

   ```
   Bob's price per acorn = $6000 ÷ b.
   Alice's total payment = 9 × $6000 = $54,000.
   Alice's price per acorn = $54,000 ÷ 3600 = $15. \boxed{{15}}
   ```

   error_tags: []

   rationale: The model correctly calculates Alice's price per acorn. While it mentions Bob's price per acorn calculation (not needed), the final answer is correct.

Problem:
`{question}`

Model Response:
`{model_response}`

Instructions:

1. Convert the given model response into First-Order Logic with sections: Signature (variables/constants), Formalization (First order Logic statements), Derivation (logical steps), Check (verification).
2. Compare the model's FOL against the problem requirements. Identify where the premises contradict the problem, rules are invalid, steps are missing, or arithmetic is incorrect.
3. For False Premise: Only apply if the model makes unreasonable assumptions that contradict the problem or common sense
   - Do NOT apply False Premise for reasonable inferences based on common knowledge (e.g., assuming standard furniture has typical leg counts)
   - Do NOT apply False Premise when the model correctly identifies that certain information is not needed for the solution
   - Do NOT apply False Premise when the model correctly sets up mathematical relationships and solves them properly
4. For False Rule: Only apply if the model uses an incorrect mathematical rule or operation (e.g., adding percentages incorrectly, using wrong formulas)
   - Do NOT apply False Rule if the model correctly follows the problem's mathematical requirements
5. For Calculator Error:

- Only flag if addition, subtraction, multiplication and division are incorrect (e.g., $5 \times 6 = 10$, is incorrect multiplication)
- do NOT apply calculator error tag if the addition, subtraction, multiplication and division are corr ect

6. For "None" error tag: Only return empty error_tags [] if the model response is completely correct with no errors of any type.
   - A response with correct calculations, proper format, complete steps, and accurate premises should have error_tags: []
   - Do NOT return "None" as an error tag - use an empty list [] instead
   - Only flag errors when they actually exist
7. Output ALL applicable error_tags from the allowed set, only if there are errors. Return empty list [] if there are no errors.
8. Provide a brief rationale for each error tag applied.

CRITICAL RULES:

- Use exact error labels: `"False Premise"`, `"False Rule"`, `"Missing Steps"`, `"Calculator Error"`, `"Format Error"`
- For no errors, use empty list: `[]`
- Only flag errors that actually exist
- Be conservative - when in doubt, don't flag an error
- Focus on the most obvious/clear errors first
- **CALCULATOR ERROR RULE**: Only flag if the actual math is wrong (e.g., $5 \times 6 = 10$). Do NOT flag if calculations are correct but the answer differs from expected

IMPORTANT: Return ONLY a valid JSON object. Do not include any other text before or after the JSON."""

E Error Categories description with Examples

1. **False Premise:**
   - **Conceptual Misunderstanding:** Misinterpreting the overall scenario or problem structure. Example: Reading "4 vacations per year" as "4 vacations total".
   - **Semantic Error:** Misusing terms or values that distort the intended meaning. Example: Treating a monthly salary of $600 as annual.
   - **False Assumption:** Introducing facts not supported by the problem. Example: Assuming "the pineapple drink is spilled" when only "a drink" is mentioned.
   - **Units Misinterpretation:** Confusing or conflating measurement units or quantities. Example: Interpreting a tank's total volume as water poured.
2. **False Rule:**
   - **Type / Operand Mismatch:** Operation applied to incompatible types, units, or domains. Example: Adding 10% of monthly salary to compute an annual raise.
   - **Inference Violation:** Conclusion does not logically follow from valid premises. Example: From "some cats are black," infer "all black things are cats."
   - **Operation Misapplication:** The operation or rule itself is inappropriate for the problem context. Example: Using compound interest for a one-time insurance fee.
   - **Quantifier Misuse:** Misplacing or misinterpreting logical quantifiers $\forall$ and $\exists$, causing overgeneralization or unwarranted restriction. Example: Interpreting "each" as "all at once."
   - **Missing necessary steps:** Key reasoning or computation steps are skipped, breaking the logical chain to reach a valid conclusion.
3. **Calculator Error:**

- Simple numeric or arithmetic mistakes **only if the numeric calculation is mathematically incorrect**.
- Examples: $5 \times 6 = 10$ instead of 30; miscomputing $7.5 + 2.5 = 11$; $9 \div 3 = 4$ instead of 3.

Note. The judge is prompted for Missing Steps and Format errors. We do not count Missing Steps separately for trace coherence but is counted under False Rule, as in Table 1. Further, Format Error is used to locate formatting problems in a response, but it is not used in the trace-coherence calculation. Whether the final answer is correctly formatted is checked by the verifier as part of answer correctness. A trace is labeled coherent or incoherent from the three categories alone (False Premise, False Rule (along with Missing Steps) and Calculator Error).