LEARNING FORMAL-INFORMAL ROUTING FOR STEP-WISE COT VERIFICATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Large language models (LLMs) often solve complex reasoning tasks by generating chain of thought (CoT), making step-level verification critical. Existing approaches rely on either *formal verification*, whose effectiveness depends on whether a CoT can be faithfully formalized and whether the resulting formal representation preserves its meaning, or *informal LLM-based verification*, which is prone to inconsistent judgements. We propose an adaptive routing policy framework that learns to select the appropriate verification tool (*formal* or *informal*) for each CoT verification instance. The policy is initialized through supervised fine-tuning (SFT) and subsequently optimized with reinforcement learning (RL) using two latent faithfulness scores, which encourage it to choose formal verifier only when symbolic formal translation is sufficiently faithful and semantically consistent.

We evaluate the framework on six reasoning benchmarks, including SIEVE, a new step-level verification dataset introduced in this work to assess generalization. Across all benchmarks, our RL routing policy outperforms 1) neuro-symbolic pipelines (VANESSA, PARC-GRAPH) and the formal-only verifier, is competitive with the 2) informal-only verifier, and rivals or exceeds substantially larger 3) proprietary verifiers such as GPT-4.1 and GPT-5.4. The improvement reaches ~ 36 Macro-F1 points over the formal-only verifier and up to ~ 11 points over the informal-only verifier. Its edge over the larger proprietary verifiers is clearest on the challenging SIEVE and PRM benchmarks, where the RL routing policy attains the best results among comparable systems and surpasses even GPT-4.1 and GPT-5.4 on SIEVE (e.g., 59.51 vs. $53.12/34.18$ Macro-F1 on SIEVE-MCQ) and exceeds GPT-4.1 on PRM (58.00 vs. 55.73). These results demonstrate that adaptive verification tool selection is an effective strategy for improving step-level reasoning verification.

1 INTRODUCTION

Chain-of-thought (CoT) prompting (Wei et al., 2022) has become widely used to elicit complex reasoning from large language models (LLMs). By generating intermediate steps, researchers have shown that LLMs can solve challenging multi-hop, logical, and commonsense reasoning tasks. Recent reasoning-focused models such as OpenAI o1 (OpenAI et al., 2026) and DeepSeek-R1 (Guo et al., 2025) have further demonstrated the potential of extended reasoning traces. However, researchers have also shown that long reasoning chains do not guarantee logical validity: LLMs may produce incorrect intermediate steps or inconsistent arguments even when the final answer is correct (Turpin et al., 2023). Therefore, applications requiring trustworthy reasoning, such as scientific analysis, legal reasoning, and education, must verify individual reasoning steps.

To obtain trustworthy verification (and reasoning), recent approaches translate each reasoning step into a formal language and check it with a logical solver, which provides soundness guarantees when the translation is faithful (Feng et al., 2025; Pan et al., 2023a; Olausson et al., 2023). However, researchers face three key challenges in formal verification: 1) some sentences lack faithful formal equivalents because of limited expressivity of the target logical languages, 2) automatic translation remains error-prone even for strong LLMs (Olausson et al., 2023; Wu et al., 2022), and 3) some claims remain difficult to verify because they depend on subjective judgments or context. These

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.

challenges are particularly evident in commonsense reasoning, which involves modality (reasoning over time), generic statements, causal and counterfactual claims, vague or graded predicates, and subjective expressions that lack faithful first-order representations[1]. Consider '*Maria enjoyed the lecture*": its subjective nature prevents a solver from deriving its truth unless the claim appears explicitly in the premises. A single step can also mix verifiable and unverifiable content: in '*the novel that Maria admired was written by a Nobel laureate*", a solver can verify the factual claim (*written by a Nobel laureate*) but cannot faithfully represent the subjective claim (*Maria admired*). Moreover, supporting contexts may contain irrelevant or unverifiable sentences, increasing translation effort and opportunities for error. Thus, the reliability of symbolic verification varies across reasoning steps, making indiscriminate formal verification potentially counterproductive.

More broadly, existing CoT verification approaches rely on either informal language-based or formal symbolic verification. Informal approaches, including LLM-as-a-judge methods (Mukherjee et al., 2025; Jacovi et al., 2024a) and NLI-based systems (Chen et al., 2021), directly assess whether a reasoning step follows from its context and remain flexible across diverse natural language inputs. However, they can struggle with complex logical dependencies and subtle contradictions. Formal approaches (Sadeddine & Suchanek, 2025; Ling et al., 2023; Pan et al., 2023a; Olausson et al., 2023), in contrast, translate reasoning steps into symbolic representations and use logical solvers to provide stronger guarantees, but their reliability depends on accurate formalization. Translation errors can therefore lead to incorrect verification decisions. Consequently, neither strategy works reliably for every reasoning instance: some instances benefit from symbolic verification, while others require the flexibility of language-based verification. This raises a fundamental question: *how can a verification system automatically determine which verification strategy is most appropriate for each reasoning instance?*

In this work, we propose an adaptive routing policy framework that formulates reasoning verification as a strategy selection problem. Instead of applying a fixed verification pipeline, our framework learns to route each verification instance among formal, and informal strategies. To guide this routing process, we introduce two latent faithfulness signals that capture complementary aspects of symbolic verification reliability: whether an instance can be faithfully represented in a formal language and whether the resulting symbolic representation preserves the original meaning. We incorporate these signals into a constraint-based reinforcement learning objective, allowing the routing policy to learn verification strategies that maximize verification accuracy while avoiding unreliable verification choices.

To evaluate the generalization capability of our routing policy leading to *adaptive verification*, we use the PRM880K (Lightman et al., 2023) dataset and additionally introduce a new benchmark dataset SIEVE (**S**tep-w**I**s**E** **V**erification of r**E**asoning) for step-level chain-of-thought verification derived from the reasoning traces of the CHECKWHY (Si et al., 2024) and FAITHCOT-BENCH (Shen et al., 2026) datasets. Unlike existing synthetic reasoning benchmarks, our dataset contains diverse reasoning patterns and linguistic variations, enabling a more realistic evaluation of verification systems. We conduct extensive experiments on SIEVE and five benchmark families, including EntailmentBank (Dalvi et al., 2021), ProofWriter (Tafjord et al., 2021), FOLIO (Han et al., 2024), ProntoQA (Saparov & He, 2023), and PRM880K. Our results demonstrate that the proposed routing framework consistently improves over existing verification baselines including formal-only, informal-only, and neuro-symbolic frameworks, and remains competitive with substantially larger proprietary models.

Our contributions are summarized as follows:

- We introduce an adaptive verification routing policy framework that learns to route reasoning instances among *formal*, and *informal* verification strategies.
- We propose a constraint-based reinforcement learning formulation guided by latent faithfulness signals, enabling the policy to learn when symbolic verification is reliable and when informal verification strategies are preferable.
- We introduce a new step-level chain-of-thought verification dataset SIEVE constructed from naturally occurring reasoning traces of CHECKWHY and FAITHCOT-BENCH datasets creating QA and MCQ variants to support evaluation under diverse reasoning conditions.

[1]While advanced logical languages can represent some of these phenomena, no single logical language can faithfully represent all natural language phenomena.

- We perform extensive experiments across SIEVE and five other logical and commonsense reasoning benchmarks, demonstrating improvements over existing stand-alone verification strategies, neuro-symbolic verification systems, and strong proprietary verification models.

2 RELATED WORK

Step-level chain-of-thought verification. A growing body of work focuses on evaluating reasoning traces at the step level rather than relying only on final-answer correctness. Process reward models (Lightman et al., 2024) learn to assess the validity of individual reasoning steps using human supervision, while the REVEAL benchmark (Jacovi et al., 2024a) provides fine-grained evaluation of step relevance, attribution, and logical correctness. Other approaches perform deductive verification by checking whether each reasoning step follows from previously established information. For example, Ling et al. (2023) propose converting reasoning steps into explicit logical relations to enable step-wise validity checking. More recent systems such as VANESSA (Sadeddine & Suchanek, 2025) combine language models with symbolic reasoning to verify individual reasoning steps under different perturbation settings. While these approaches improve the reliability of reasoning verification, these methods employ fixed verification strategies, whereas our work studies adaptive verification strategy selection.

Neuro-symbolic verification. Neuro-symbolic approaches combine the flexibility of language models with the logical guarantees of symbolic reasoning. Several works translate natural language reasoning problems into formal representations and employ external solvers for verification or inference. Logic-LM (Pan et al., 2023a) uses LLMs to generate symbolic programs that are executed by logical solvers, while LINC (Olausson et al., 2023) explores translating reasoning problems into symbolic forms for theorem proving. Similarly, verification frameworks such as VERICOT (Feng et al., 2025) use formalization and theorem proving to analyze the logical consistency of chain-of-thought reasoning. Although these approaches provide stronger logical guarantees, their performance depends on faithful formalization and symbolic translation. Existing neuro-symbolic systems typically employ predetermined verification pipelines, motivating adaptive verification strategies that can exploit the strengths of complementary verification mechanisms. This perspective is related to adaptive computation and routing methods (Shazeer et al., 2017; Fedus et al., 2022), but differs in that our routing objective focuses on selecting verification strategies according to reasoning reliability rather than computational efficiency.

Benchmarks for reasoning verification. Several datasets have been introduced to evaluate logical reasoning and chain-of-thought verification. ProofWriter provides synthetic rule-based reasoning problems with explicit logical structures, while EntailmentBank evaluates multi-step entailment reasoning through structured proof trees. These benchmarks have enabled systematic evaluation of deductive reasoning systems but primarily contain synthetic or highly structured reasoning traces. In contrast, CHECKWHY (Si et al., 2024) introduces a fact verification benchmark containing argumentative reasoning chains extracted from real-world text, capturing greater linguistic diversity and naturally occurring reasoning patterns. More recently, FAITHCOT-BENCH (Shen et al., 2026) introduces the benchmark for instance-level chain-of-thought faithfulness detection, comprising expert-annotated reasoning trajectories with faithfulness labels, fine-grained error categories, and supporting evidence. Motivated by the need for realistic evaluation of step-level verification, we construct our new verification dataset from both the CHECKWHY and the FAITHCOT-BENCH by extending it with annotations and evaluation settings tailored for reasoning chain verification.

3 PROPOSED FRAMEWORK

3.1 PROBLEM FORMULATION

Given a reasoning problem consisting of a question Q, a context C, and a chain of thought $\mathcal{S} = \{s_1, s_2, \ldots, s_T\}$, our goal is to verify the correctness of every reasoning step $s_i, 1 \leq i \leq T$. Here, the step s_i can be either an *attribution step* (which requires external evidence) or a *logical step* (which can be derived from previous steps in the chain of thought). We convert each reasoning chain into a collection of independent verification instances. Each verification instance is

represented as $x = (P, H)$, where the hypothesis H corresponds to a single reasoning step s_i to be verified and P denotes its supporting premises. Depending on the type of reasoning step, the premises are constructed differently. For *attribution steps*, the premises consist of the corresponding evidence paragraph whenever such evidence is available. For *logical steps*, the premises consist of the original context together with all previously verified reasoning steps. Consequently, every verification instance contains exactly one hypothesis but different premise sets. Each verification instance is associated with a gold verification label $y^* \in \{\texttt{follows}, \texttt{not-follows}\}$.

3.2 Routing Framework

The proposed routing framework consists of two components: (1) a routing policy that selects an appropriate verification tool for each verification instance, and (2) a collection of complementary verification tools, i.e., *formal*, and *informal* tools. The routing policy operates over the discrete action space $\mathcal{A} = \{a_F,\ a_I\}$, corresponding to the **Formal**, and **Informal** verification tools.

Formal Verification Tool. The formal tool jointly translates the verification instance $x = (P, H)$ into SMT-LIB v2.6 (Barrett et al., 2016) form using an LLM and then verifies logical entailment with the Z3 theorem prover (De Moura & Bjørner, 2008). The Z3 theorem prover performs consistency, entailment, and contradiction checks before returning the final verdict. This tool provides exact logical guarantees whenever the translation faithfully preserves the original semantics. However, its performance depends critically on whether the verification instance can be represented accurately within the expressive limits of SMT-LIB. The prompt to translate the verification instance into SMT-LIB v2.6 is provided in the appendix B.1.

Informal Verification Tool. We use a verification prompt for the informal tool to directly verifies the instance $x = (P, H)$ using a Large Language Model. Unlike the formal tool, it does not require symbolic translation and therefore remains applicable even when the reasoning contains linguistic phenomena that cannot be represented in first-order logic. Its predictions, however, rely entirely on the language model and may therefore contain logical errors. We provide the verification prompt in the appendix B.2.

3.3 Latent Faithfulness Scores

A key challenge in using the formal verification tool is determining whether a verification instance $x = (P, H)$ can be translated into symbolic logic without altering its semantic meaning. Instead of assuming that every reasoning step is equally suitable for formal reasoning, our framework estimates two latent faithfulness scores that quantify the reliability of symbolic verification before the routing policy is optimized. These scores are computed over the complete verification instance $(x = P, H)$, and are incorporated into the reinforcement learning objective through Lagrangian constraints (3.4), encouraging the routing policy to invoke formal verification only when symbolic reasoning is expected to be both applicable and semantically reliable.

Formalizability Score (Z_F). The first latent score estimates whether a verification instance can be *faithfully represented* within the SMT-LIB fragment supported by Z3 parser. Rather than attempting formalization directly, an LLM analyzes the premise passage and the hypothesis independently to identify semantic phenomena that influence symbolic expressibility. These include six phenomena that are **readily expressible** in first-order logic (universal quantification, existential quantification, logical negation, arithmetic/counting, binary predicates and temporal ordering) and six phenomena that are **not faithfully expressible** (modal meaning, generic statements, counterfactuals, causal claims, performative utterances, and gradable expressions without explicit thresholds)). For every non-expressible phenomenon, the LLM additionally determines whether it is *central* or *peripheral* to the meaning of the passage, with central phenomena receiving a larger penalty. This deterministic scoring procedure produces formalizability scores $z_P^F, z_H^F \in [0, 1]$ for the premise block and hypothesis, respectively. The instance-level score is then computed using a weakest-link aggregation, $Z_F = \min\big(z_P^F,\ z_H^F\big)$, reflecting the intuition that successful symbolic verification requires both the supporting premises and the hypothesis to admit faithful logical representations. Appendix B.3 and C.2 contains the prompt and the algorithm used to compute the score Z_F respectively.

Semantic Consistency Score (Z_C). Even when a verification instance is sufficiently formalizable, the generated SMT-LIB representation may still omit or distort important semantic information during translation. To estimate this translation fidelity, we compute a semantic consistency score using *backtranslation*. The SMT-LIB representations of the premise block and the hypothesis are independently translated back into natural language using an LLM. Sentence-BERT (`all-MiniLM-L6-v2`) (Reimers & Gurevych, 2019) then measures the semantic similarity between each original natural langugage text block and its corresponding backtranslated version, producing scores $z_P^C, z_H^C \in [0, 1]$. The instance-level consistency score is computed as the geometric mean, $Z_C = \sqrt{z_P^C \times z_H^C}$, which naturally penalizes cases where either the premise representation or the hypothesis representation undergoes substantial semantic drift during symbolic translation. Appendix B.4 and C.3 contains the prompt and the algorithm used to compute the score Z_C respectively.

3.4 Routing Policy

Our routing policy learns a mapping from a verification instance $x = (P, H)$ to one of the two verification tools, $\pi_\theta(a \mid x)$, $a \in \mathcal{A} = \{a_F, a_I\}$, where a_F and a_I denote the action corresponding to the formal, and informal tools, respectively. Importantly, the routing policy only selects the verification tool given the reasoning instance $x = (P, H)$. The verification label $\hat{y}$ is produced by the tool selected by the policy, and its correctness determines the reward, that used to optimize the routing policy. Specificially, the routing policy is optimized to maximize downstream verification accuracy (whether the the selected tool returns the correct verdict $\hat{y} = y^*$), subject to faithfulness constraints on formal verification. Rather than directly predicting the route action token, the policy generates a short reasoning trace followed by the selected tool action. The generated output has the form $\langle$`think`$\rangle$ c $\langle$`/think`$\rangle$ $\langle$`route`$\rangle$ a $\langle$`/route`$\rangle$, where c is the model's routing rationale and $a \in \{a_F, a_I\}$ is the selected verification tool. Only the generated completion contributes to the training objective, while the prompt is masked. The routing policy is trained in two stages. We first obtain an initial *warm-up* policy using supervised fine-tuning and then further optimize it using REINFORCE Leave-One-Out (RLOO) policy optimization algorithm (Kool et al., 2019).

3.4.1 Supervised Fine-Tuning

We initially use a supervised finetuning (SFT) step to learn a *warm-up* policy before RL stage. For this stage, we construct routing supervision by running both tools on each training instance and comparing each tool's predicted label to gold verification label $y^* \in \{\texttt{follows}, \texttt{not-follows}\}$. We select the formal tool whenever its (Z3-based) verdict matches y^*; otherwise, we select the informal tool if the predicted label is correct. We exclude instances from supervised fine-tuning step, if neither tool is correct. Instead, they are handled during the subsequent RL stage. Details of the training data construction algorithm, prompts, the source datasets for the training set, and statistics of the resulting supervised training dataset are provided in the appendix D.1. In summary, the training set consists of N (instance, oracle-tool) pairs balanced across the two verification tools. The SFT policy is optimized using the standard autoregressive cross-entropy objective $\mathcal{L}_{\text{SFT}} = -\sum_{t=1}^{|y|} \log \pi_\theta(y_t \mid y_{<t}, x)$, where $y = (y_1, \ldots, y_{|y|})$ is the generated completion, comprising the tokens of the `<think>` and `<route>` fields, and the prompt x is masked out of the loss.

3.4.2 Reinforcement Learning

In the SFT step, the policy learns to predict pre-assigned route labels using cross-entropy rather than optimizing the downstream objective stated in Section 3.4—namely, maximizing whether the selected tool's verdict label $\hat{y}$ matches the gold label y^*, under constraints such as verification cost and faithful formal translation. Since quantifying the above aspects require online tool-execution, we therefore turn to a policy-gradient based method. We use REINFORCE Leave-One-Out (RLOO) (Kool et al., 2019), which fits our setting.

RLOO Policy Optimization. For every input instance, RLOO samples G trajectories. Instead of learning a separate value function, RLOO estimates the advantage of the g-th trajectory using the average reward of the remaining samples from the same input, $A^{(g)} = R^{(g)} - \frac{1}{G-1} \sum_{j \neq g} R^{(j)}$. The

routing policy is optimized using the policy-gradient objective $\mathcal{L}_{\mathrm{PG}} = -\mathbb{E}\left[A^{(g)} \log \pi_\theta(a^{(g)}|x)\right]$, together with a KL regularization term that prevents the policy from drifting excessively from the supervised initialization, $\mathcal{L} = \mathcal{L}_{\mathrm{PG}} + \beta D_{\mathrm{KL}}(\pi_\theta \| \pi_{\mathrm{SFT}})$. Since, our routing policy selects a single verification tool per instance and receives one outcome reward for that choice, there is no credit assignment across steps and no need for a learned value function. With only two tools to choose from, sampling a few routing trajectories per instance and scoring each by its verification reward is cheap, and RLOO's leave-one-out baseline yields a low-variance advantage estimate without training a separate value network.

Reward Function. For each training instance x, the policy samples G independent routing trajectories $\tau^{(g)} = (c^{(g)}, a^{(g)})$, $g = 1, \ldots, G$, where $c^{(g)}$ denotes the generated routing rationale and $a^{(g)}$ denotes the selected verification tool. Each selected tool is executed to produce a verdict $\hat{y}^{(g)}$, which is mapped to a scalar reward $R(x, a^{(g)})$ in the following way. For every sampled trajectory, the policy receives the reward (dropping the superscript for readability)

$$R = \begin{cases} +\delta, & a \neq \varnothing \wedge \hat{y} \neq \varnothing, \\ -\delta, & \text{otherwise}, \end{cases} + \mathbf{1}[\hat{y} = y^*] \quad -\lambda c_a - \mathbb{1}(a = \textsc{Formal})\Big[\mu_F(1 - Z_F) + \mu_C(1 - Z_C)\Big]. \tag{1}$$

where δ is the formatting bonus. The policy receives $+\delta$ if it successfully generates both a valid verification tool a and a corresponding verification prediction $\hat{y}$, and $-\delta$ otherwise. Here, $\hat{y}$ denotes the verifier prediction, y^* is the gold verification label, c_a is the computational cost of the selected verification tool, and λ controls the cost penalty. The function $\mathbb{1}(a = \textsc{Formal})$ equals 1 for the formal verification tool (0 otherwise), to apply the latent-faithfulness penalties only to the formal tool that relies on symbolic reasoning.

Formally, we solve

$$\max_\theta \ \mathbb{E}_{x \sim \mathcal{D}_{\mathrm{RL}},\, a \sim \pi_\theta(\cdot|x)}[R(x, a) - \beta D_{\mathrm{KL}}(\pi_\theta \| \pi_{\mathrm{SFT}})] \tag{2}$$

subject to $m_F \leq \varepsilon$, $m_C \leq \varepsilon$, where π_θ denotes the routing policy parameterized by θ, $\mathcal{D}_{\mathrm{RL}}$ is the reinforcement learning training dataset, and a is the verification route sampled from the policy. Here, m_F and m_C quantify how often the policy routes to the formal tool on instances that the latent faithfulness scores of Section 3.3 deem unreliable: m_F averages $(1 - Z_F)$ and m_C averages $(1 - Z_C)$ over the sampled trajectories that select the formal route (defined precisely in Section 3.4.3, Eqs. equation 3–equation 4). A larger $(1 - Z_F)$ or $(1 - Z_C)$ indicates that the instance is less formalizable or that its symbolic translation is less faithful, so m_F and m_C grow precisely when formal verification is invoked where Z_F or Z_C judge it unreliable.

3.4.3 Constraint Statistics and Dual Optimization

The latent faithfulness scores are incorporated into the optimization through adaptive Lagrangian penalties rather than fixed regularization coefficients. After every policy update, the algorithm estimates how frequently the policy selects verification tools that are poorly supported by the latent faithfulness scores. For a minibatch containing B problem instances and G sampled routing trajectories per instance, we define the average formal-tool misuse as

$$m_F = \frac{1}{BG} \sum_{b=1}^{B} \sum_{g=1}^{G} \left(1 - Z_F^{(b)}\right) \mathbf{1}[a_{b,g} = \textsc{Formal}], \tag{3}$$

$$m_C = \frac{1}{BG} \sum_{b=1}^{B} \sum_{g=1}^{G} \left(1 - Z_C^{(b)}\right) \mathbf{1}[a_{b,g} = \textsc{Formal}]. \tag{4}$$

Intuitively, m_F measures how often the policy invokes symbolic reasoning on instances that are unlikely to admit a faithful formalization, whereas m_C measures how frequently symbolic reasoning is selected despite weak semantic consistency between the natural-language statements and their generated symbolic representations. The corresponding dual variables are updated after each minibatch using projected gradient ascent,

$$\mu_F \leftarrow \mathrm{clip}(\mu_F + \eta_\mu(m_F - \varepsilon),\, 0,\, \mu_{\max}), \tag{5}$$

$$\mu_C \leftarrow \mathrm{clip}(\mu_C + \eta_\mu(m_C - \varepsilon),\, 0,\, \mu_{\max}), \tag{6}$$

where η_μ is the dual learning rate, ε is the desired tolerance for formal-tool misuse, and clip opeator projects the updated dual variable onto the interval $[0, \mu_{\max}]$, preventing it from becoming negative or growing without bound. Consequently, whenever the observed misuse rate exceeds the target tolerance, the corresponding dual variable increases, making low-faithfulness formal verification progressively more expensive in subsequent policy updates. Conversely, if the misuse rate falls below the target threshold, the penalty decreases, allowing the policy to use the formal verification tool whenever it is supported by the latent faithfulness estimates. We use `Qwen-2.5-7B-Instruct` model for both SFT and RL routing policy, and `GPT-4.1-mini, GPT-OSS-20B` models for both formal and informal verification tools. Section E of appendix provides the complete hyperparameter configuration for the reinforcement learning stage, together with detailed statistics of the training dataset.

4 SIEVE: A DATASET FOR STEP-WISE VERIFICATION OF REASONING

Construction. We construct a step-level verification dataset SIEVE from CHECKWHY and FAITHCOT-BENCH dataset as follows: 1) For each claim in the CHECKWHY dataset, we prompt an LLM to generate reasoning CoT chains and annotate each step as either an *attribution step* or a *logical step*. We then partition each reasoning chain at every logical step, producing multiple progressively extended instances in which each instance terminates at a logical inference point. We refer to this resulting dataset as the first variant of the SIEVE dataset, namely SIEVE-QA, which contains 1,228 instances, 2) From FAITHCOT-BENCH, we use the chain-of-thought reasoning traces corresponding to the Logic-QA (Liu et al., 2020), TruthfulQA (Lin et al., 2022), and HLE (and Phan et al., 2026) benchmarks to construct the second variant, SIEVE-MCQ. SIEVE-MCQ containing 950 instances in total. For both datasets, each verification instance is independently annotated by two human annotators with a binary `follows`/`not-follows` label indicating whether the hypothesis follows from its supporting premises. Cohen's κ is 0.70 for SIEVE-QA and 0.64 for SIEVE-MCQ.

Comparative Analysis Table 6 in appendix presents a comparison of reasoning benchmarks based on context type, human annotation availability, training split release, and CoT source. SIEVE differs from prior benchmarks by supporting mixed context sources and incorporating human-annotated labels alongside diverse CoT sources for step-level verification. Section F of appendix contains additional details about the label distribution, comparison of token length, sentence count, exact-match duplication rate, and information entropy of the SIEVE-QA variant dataset with other reasoning verification benchmarks.

5 EXPERIMENTAL SETTING

This section evaluates the proposed adaptive routing policy framework across multiple reasoning benchmarks. We first describe the datasets, and baseline verification tools. We then present a comparison against those baselines. All quantitative results are reported using the Macro-F1 score.

5.1 DATASETS

We evaluate our framework on six benchmark families covering diverse reasoning settings in chain-of-thought verification. For four existing benchmark families, we use the perturbed variants released by VANESSA (Sadeddine & Suchanek, 2025). **EntailmentBank** contains multi-step reasoning problems where entailment trees connect facts to final hypotheses through intermediate conclusions; we evaluate its *Negation* and *Hallucination* variants released by VANESSA. **ProofWriter** evaluates deductive reasoning over synthetic rule-based theories, and we consider its *Negation*, *Hallucination*, and *Remove* variants. **FOLIO** provides natural language reasoning problems with first-order logic annotations, while **PrOntoQA** focuses on compositional multi-hop reasoning over synthetic rule-based environments; for both, we evaluate the traces generated by the *LLaMA-2*, *LLaMA-3*, and *Mixtral* models under the VANESSA perturbation protocol. Finally, to check the generalization capabilities, we evaluate on **PRM800K** dataset, and our proposed **SIEVE** dataset. Collectively, these benchmarks cover both natural and synthetic reasoning tasks, varying levels of logical complexity, and reasoning traces generated by multiple language models. This diversity allows us to evaluate

Table 1: Macro-F1 score (%) on EntailmentBank (**EBank**), SIEVE, PRM, FOLIO, and PRONTOQA. Here, **Verifier LLM** represents the model used as both Formal and Informal Verification tool, while **Method** represents the different baselines, and approaches. Our proposed RL Policy approach uses `Qwen-2.5-7B-Instruct` for routing. **Bold** and underline mark the best and second-best scores, respectively, among the GPT-4.1-mini and GPT-OSS-20B rows; GPT-5.4 and GPT-4.1 are reported as reference (near-upper-bound) baselines and are excluded from this comparison. A dash (–) marks an (LLM, method) pair not evaluated on that dataset.

Verifier LLM	Method	EBank		SIEVE		PRM	FOLIO			PRONTOQA		
		Neg	Hallu	QA	MCQ	PRM	LLaMA2	LLaMA3	Mixtral	LLaMA2	LLaMA3	Mixtral
GPT-5.4	Informal Only	88.04	80.82	61.15	34.18	59.28	84.81	79.06	83.27	95.48	93.75	91.99
GPT-4.1	VANESSA	72.51	68.30	–	–	–	–	–	–	–	–	–
	PARC-GRAPH	76.48	69.87	55.50	–	–	61.44	61.46	62.85	65.42	65.49	66.34
	Informal Only	89.55	83.32	61.06	53.12	55.73	82.94	80.58	79.61	91.35	97.40	86.65
GPT-4.1 -mini	Formal Only	45.85	30.49	21.08	26.05	29.19	53.44	44.60	42.59	77.79	78.53	74.39
	Informal Only	85.97	**78.22**	58.19	48.06	51.08	77.46	70.15	74.16	83.95	90.09	**89.98**
	RL Policy	**86.99**	62.85	**65.66**	**59.51**	51.76	73.79	73.18	69.92	88.01	91.83	80.95
GPT-OSS -20B	Formal Only	50.15	48.58	22.08	27.31	31.13	69.21	55.22	64.31	79.43	67.46	63.10
	RL Policy	86.23	72.57	58.59	–	**58.00**	**85.10**	**82.15**	**80.06**	**90.47**	**92.91**	84.14

whether the proposed routing policy generalizes across different domains, reasoning styles, and error distributions.

5.2 BASELINES

We compare the proposed framework against the following categories of baseline verification tools.

Neuro-symbolic Verification Tools. We compare against neuro-symbolic baselines, including VANESSA (Sadeddine & Suchanek, 2025) and a **Formal-only** verifier. VANESSA uses `GPT-4.1` as the verifier model. For the formal-only verifier, we first translate the verification instance (P, H) into an SMT-LIB form using various verifier models like `GPT-4.1-mini, GPT-OSS-20B` and then use the Z3 solver to determine whether the hypothesis follows from the given context.

Informal-only Verification Tools. We compare against informal-only verification baselines, including standalone proprietary LLM verifiers such as `GPT-5.4`, `GPT-4.1`, and the graph-based verifier built upon PARC (Mukherjee et al., 2025). Specifically, we adopt PARC's Aggregative Premise Mapping strategy to select premises relevant to the given hypothesis and complemented it with our novel graph-based verification approach described in section G of appendix. We use `GPT-4.1` as the underlying verifier model for PARC-GRAPH.

6 MAIN RESULTS

Tables 1, 2 report the performance of the routing framework across six benchmark. Across all benchmarks, several consistent observations emerge:

Comparison with neuro-symbolic baselines. RL routing policies decisively outperform the neuro-symbolic verification baselines VANESSA and PARC-GRAPH, despite those baselines using the substantially stronger GPT-4.1 as their verifier model. Against PARC-GRAPH the margins are large and consistent: RL policy (using GPT-4.1-mini as verifier LLM) raises ProofWriter-Neg from 58.65 to 98.40 (+39.75), and the RL policy (using GPT-OSS-20B as verifier LLM) lifts all the three FOLIO splits from $\sim$61–63 to 80–85 and PRONTOQA LLaMA2 from 65.42 to 90.47. VANESSA trails our mini policy (88.83 vs. 98.40 on Neg) on ProofWriter dataset. We didn't show VANESSA results on FOLIO and ProntoQA, where dozens of examples triggered content-filtering violations under `GPT-4.1`, and on SIEVE and PRM, where the approach is prohibitively time-consuming. Our lightweight router (Qwen-2.5-7B-Instruct) over a formal/informal tool pair thus far exceeds a fixed symbolic pipeline, and the advantage does not stem from a larger verifier LLM.

Comparison with larger verifiers. More notably, our lightweight router using smaller GPT-4.1-mini and GPT-OSS-20B verifer models match or exceed the near-upper-bound closed models on several datasets. On SIEVE, the GPT-4.1-mini based RL-policy surpasses both references by a wide margin—59.51 on SIEVE-MCQ against 34.18 (GPT-5.4) and 53.12 (GPT-4.1), and 65.66 on SIEVE-QA against 61.15 (GPT-5.4), 61.06 (GPT-4.1). The GPT-OSS-20B based RL policy beats GPT-4.1 on all three FOLIO splits and on PRM (58.00 vs. 55.73), and edges past GPT-5.4 on FOLIO-LLaMA2 (85.10 vs. 84.81), FOLIO-LLaMA3 (82.15 vs. 79.06) and ProofWriter-Remove (95.22 vs. 91.33). The references stay ahead on EntailmentBank and most of PRONTOQA, so routing does not close gap on every dataset—but it makes an open verifier model competitive with, and sometimes better than, frontier closed source models.

Generalization to SIEVE and PRM. The generalization results on both variants of the SIEVE and PRM datasets shows the robustness our learned RL routing policies. The formal tool nearly collapses on them (e.g., 21.08/26.05 on SIEVE and 29.19 on PRM for GPT-4.1-mini), yet the GPT-4.1-mini based RL policy improves over its own informal-only verifier by $+7.47$ (QA) and $+11.45$ (MCQ), and the GPT-OSS-20B based RL policy improves over formal-only by $+36.51$ on SIEVE-QA and $+26.87$ on PRM.

Does the policy adapt routing to the verifier? A central question is whether the policy learns *which* verification tool to trust as the verifier LLM changes. At the aggregate level, it clearly does. On identical instances the two policies route very differently: the RL policy (using GPT-4.1-mini as verifier LLM) sends only $\sim 3.5\%$ of steps to the formal route, whereas RL policy (using GPT-OSS-20B as verifier LLM) sends $\sim 23.8\%$ which is a $7\times$ difference. We averaged the performance of **Formal-only** verifier over the datasets mentioned in Table 1: formal-only macro-F1 is 58.9 for GPT-4.1-mini based RL policy and 63.7 for GPT-OSS-20B based RL policy. In the **Formal Only** setting a model must use the formal tool on every step, so this gap means GPT-OSS-20B writes more usable SMT-LIB than GPT-4.1-mini on average which results in large performance differences in the **Formal-Only** method on various dataset: difference reaches $+21.72$ points on FOLIO-Mixtral (64.31 vs. 42.59) and $+18.09$ on EntailmentBank-Hallu (48.58 vs. 30.49), the datasets where the GPT-4.1-mini based Formal only tool is weakest. The two routing rates (3.5% vs. 23.8%) mirror this reliability: each policy uses the formal tool roughly in proportion to how well that tool works for its verifier.

7 CONCLUSION

We presented an adaptive routing framework for step-level verification that learns, per instance, whether to use a formal symbolic verifier or an informal LLM verifier, since the better choice depends on the characterstics of the reasoning instance. We cast verifier selection as a routing problem and train the policy with reinforcement learning guided by two latent faithfulness signals capturing symbolic-translation quality and semantic consistency. Across EntailmentBank, ProofWriter, FOLIO, ProntoQA, PRM, and the newly introduced SIEVE benchmark, the learned RL policy improves over existing baselines and rivals substantially larger proprietary verifiers while using only a lightweight open-source router and a smaller verifier. Our results suggest that the primary challenge in reasoning verification is not identifying a universally superior verifier, but learning when each verification paradigm should be trusted. In future, we plan to extend our paradigm to include more tools, and multiple tool calls for each step.

AI USE STATEMENT

We have not used generative AI tools in the preparation of this work, including for research, analysis, writing, or coding.

REFERENCES

Long and Phan, Alice Gatti, Nathaniel Li, Adam Khoja, Ryan Kim, Richard Ren, Jason Hausenloy, Oliver Zhang, Mantas Mazeika, Dan Hendrycks, Ziwen Han, Josephina Hu, Hugh Zhang, Chen Bo Calvin Zhang, Mohamed Shaaban, John Ling, Sean Shi, Michael Choi, Anish Agrawal, Arnav Chopra, Aakaash Nattanmai, Gordon McKellips, Anish Cheraku, Asim Suhail, Ethan Luo, Marvin Deng, Jason Luo, Ashley Zhang, Kavin Jindel, Jay Paek, Kasper Halevy, Allen

Baranov, Michael Liu, Advaith Avadhanam, David Zhang, Vincent Cheng, Brad Ma, Evan Fu, Liam Do, Joshua Lass, Hubert Yang, Surya Sunkari, Vishruth Bharath, Violet Ai, James Leung, Rishit Agrawal, Alan Zhou, Kevin Chen, Tejas Kalpathi, Ziqi Xu, Gavin Wang, Tyler Xiao, Erik Maung, Sam Lee, Ryan Yang, Roy Yue, Ben Zhao, Julia Yoon, Xiangwan Sun, Aryan Singh, Clark Peng, Tyler Osbey, Taozhi Wang, Daryl Echeazu, Timothy Wu, Spandan Patel, Vidhi Kulkarni, Vijaykaarti Sundarapandiyan, Andrew Le, Zafir Nasim, Srikar Yalam, Ritesh Kasamsetty, Soham Samal, David Sun, Nihar Shah, Abhijeet Saha, Alex Zhang, Leon Nguyen, Laasya Nagumalli, Kaixin Wang, Aidan Wu, Anwith Telluri, Summer Yue, Alexandr Wang, Dmitry Dodonov, Tung Nguyen, Jaeho Lee, Daron Anderson, Mikhail Doroshenko, Alun Cennyth Stokes, Mobeen Mahmood, Oleksandr Pokutnyi, Oleg Iskra, Jessica P. Wang, John-Clark Levin, Mstyslav Kazakov, Fiona Feng, Steven Y. Feng, Haoran Zhao, Michael Yu, Varun Gangal, Chelsea Zou, Zihan Wang, Serguei Popov, Robert Gerbicz, Geoff Galgon, Johannes Schmitt, Will Yeadon, Yongki Lee, Scott Sauers, Alvaro Sanchez, Fabian Giska, Marc Roth, Søren Riis, Saiteja Utpala, Noah Burns, Gashaw M. Goshu, Mohinder Maheshbhai Naiya, Chidozie Agu, Zachary Giboney, Antrell Cheatom, Francesco Fournier-Facio, Sarah-Jane Crowson, Lennart Finke, Zerui Cheng, Jennifer Zampese, Ryan G. Hoerr, Mark Nandor, Hyunwoo Park, Tim Gehrunger, Jiaqi Cai, Ben McCarty, Alexis C. Garretson, Edwin Taylor, Damien Sileo, Qiuyu Ren, Usman Qazi, Lianghui Li, Jungbae Nam, John B. Wydallis, Pavel Arkhipov, Jack Wei Lun Shi, Aras Bacho, Chris G. Willcocks, Hangrui Cao, Sumeet Motwani, Emily de Oliveira Santos, Johannes Veith, Edward Vendrow, Doru Cojoc, Kengo Zenitani, Joshua Robinson, Longke Tang, Yuqi Li, Joshua Vendrow, Natanael Wildner Fraga, Vladyslav Kuchkin, Andrey Pupasov Maksimov, Pierre Marion, Denis Efremov, Jayson Lynch, Kaiqu Liang, Aleksandar Mikov, Andrew Gritsevskiy, Julien Guillod, Gözdenur Demir, Dakotah Martinez, Ben Pageler, Kevin Zhou, Saeed Soori, Ori Press, Henry Tang, Paolo Rissone, Sean R. Green, Lina Brüssel, Moon Twayana, Aymeric Dieuleveut, Joseph Marvin Imperial, Ameya Prabhu, Jinzhou Yang, Nick Crispino, Arun Rao, Dimitri Zvonkine, Gabriel Loiseau, Mikhail Kalinin, Marco Lukas, Ciprian Manolescu, Nate Stambaugh, Subrata Mishra, Tad Hogg, Carlo Bosio, Brian P. Coppola, Julian Salazar, Jaehyeok Jin, Rafael Sayous, Stefan Ivanov, Philippe Schwaller, Shaipranesh Senthilkumar, Andres M. Bran, Andres Algaba, Kelsey Van den Houte, Lynn Van Der Sypt, Brecht Verbeken, David Noever, Alexei Kopylov, Benjamin Myklebust, Bikun Li, Lisa Schut, Evgenii Zheltonozhskii, Qiaochu Yuan, Derek Lim, Richard Stanley, Tong Yang, John Maar, Julian Wykowski, Mart Oller, Anmol Sahu, Cesare Giulio Ardito, Yuzheng Hu, Ariel Ghislain Kemogne Kamdoum, Alvin Jin, Tobias Garcia Vilchis, Yuexuan Zu, Martin Lackner, James Koppel, Gongbo Sun, Daniil S. Antonenko, Steffi Chern, Bingchen Zhao, Pierrot Arsene, Joseph M. Cavanagh, Daofeng Li, Jiawei Shen, Donato Crisostomi, Wenjin Zhang, Ali Dehghan, Sergey Ivanov, David Perrella, Nurdin Kaparov, Allen Zang, Ilia Sucholutsky, Arina Kharlamova, Daniil Orel, Vladislav Poritski, Shalev Ben-David, Zachary Berger, Parker Whitfill, Michael Foster, Daniel Munro, Linh Ho, Shankar Sivarajan, Dan Bar Hava, Aleksey Kuchkin, David Holmes, Alexandra Rodriguez-Romero, Frank Sommerhage, Anji Zhang, Richard Moat, Keith Schneider, Zakayo Kazibwe, Don Clarke, Dae Hyun Kim, Felipe Meneguitti Dias, Sara Fish, Veit Elser, Tobias Kreiman, Victor Efren Guadarrama Vilchis, Immo Klose, Ujjwala Anantheswaran, Adam Zweiger, Kaivalya Rawal, Jeffery Li, Jeremy Nguyen, Nicolas Daans, Haline Heidinger, Maksim Radionov, Václav Rozhoň, Vincent Ginis, Christian Stump, Niv Cohen, Rafał Poświata, Josef Tkadlec, Alan Goldfarb, Chenguang Wang, Piotr Padlewski, Stanislaw Barzowski, Kyle Montgomery, Ryan Stendall, Jamie Tucker-Foltz, Jack Stade, T. Ryan Rogers, Tom Goertzen, Declan Grabb, Abhishek Shukla, Alan Givré, John Arnold Ambay, Archan Sen, Muhammad Fayez Aziz, Mark H. Inlow, Hao He, Ling Zhang, Younesse Kaddar, Ivar Ängquist, Yanxu Chen, Harrison K. Wang, Kalyan Ramakrishnan, Elliott Thornley, Antonio Terpin, Hailey Schoelkopf, Eric Zheng, Avishy Carmi, Ethan D. L. Brown, Kelin Zhu, Max Bartolo, Richard Wheeler, Martin Stehberger, Peter Bradshaw, JP Heimonen, Kaustubh Sridhar, Ido Akov, Jennifer Sandlin, Yury Makarychev, Joanna Tam, Hieu Hoang, David M. Cunningham, Vladimir Goryachev, Demosthenes Patramanis, Michael Krause, Andrew Redenti, David Aldous, Jesyin Lai, Shannon Coleman, Jiangnan Xu, Sangwon Lee, Ilias Magoulas, Sandy Zhao, Ning Tang, Michael K. Cohen, Orr Paradise, Jan Hendrik Kirchner, Maksym Ovchynnikov, Jason O. Matos, Adithya Shenoy, Michael Wang, Yuzhou Nie, Anna Sztyber-Betley, Paolo Faraboschi, Robin Riblet, Jonathan Crozier, Shiv Halasyamani, Shreyas Verma, Prashant Joshi, Eli Meril, Ziqiao Ma, Jérémy Andréoletti, Raghav Singhal, Jacob Platnick, Volodymyr Nevirkovets, Luke Basler, Alexander Ivanov, Seri Khoury, Nils Gustafsson, Marco Piccardo, Hamid Mostaghimi, Qijia Chen, Virendra Singh, Tran Quoc Khánh, Paul Rosu, Hannah Szlyk, Zachary Brown, Himanshu Narayan, Aline Menezes, Jonathan

Roberts, William Alley, Kunyang Sun, Arkil Patel, Max Lamparth, Anka Reuel, Linwei Xin, Hanmeng Xu, Jacob Loader, Freddie Martin, Zixuan Wang, Andrea Achilleos, Thomas Preu, Tomek Korbak, Ida Bosio, Fereshteh Kazemi, Ziye Chen, Biró Bálint, Eve J. Y. Lo, Jiaqi Wang, Maria Inês S. Nunes, Jeremiah Milbauer, M. Saiful Bari, Zihao Wang, Behzad Ansarinejad, Yewen Sun, Stephane Durand, Hossam Elgnainy, Guillaume Douville, Daniel Tordera, George Balabanian, Hew Wolff, Lynna Kvistad, Hsiaoyun Milliron, Ahmad Sakor, Murat Eron, Andrew Favre, Shailesh Shah, Xiaoxiang Zhou, Firuz Kamalov, Sherwin Abdoli, Tim Santens, Shaul Barkan, Allison Tee, Robin Zhang, Alessandro Tomasiello, G. Bruno De Luca, Shi-Zhuo Looi, Vinh-Kha Le, Noam Kolt, Jiayi Pan, Emma Rodman, Jacob Drori, Carl J. Fossum, Niklas Muennighoff, Milind Jagota, Ronak Pradeep, Honglu Fan, Jonathan Eicher, Michael Chen, Kushal Thaman, William Merrill, Moritz Firsching, Carter Harris, Stefan Ciobâcă, Jason Gross, Rohan Pandey, Ilya Gusev, Adam Jones, Shashank Agnihotri, Pavel Zhelnov, Mohammadreza Mofayezi, Alexander Piperski, David K. Zhang, Kostiantyn Dobarskyi, Roman Leventov, Ignat Soroko, Joshua Duersch, Vage Taamazyan, Andrew Ho, Wenjie Ma, William Held, Ruicheng Xian, Armel Randy Zebaze, Mohanad Mohamed, Julian Noah Leser, Michelle X. Yuan, Laila Yacar, Johannes Lengler, Katarzyna Olszewska, Claudio Di Fratta, Edson Oliveira, Joseph W. Jackson, Andy Zou, Muthu Chidambaram, Timothy Manik, Hector Haffenden, Dashiell Stander, Ali Dasouqi, Alexander Shen, Bita Golshani, David Stap, Egor Kretov, Mikalai Uzhou, Alina Borisovna Zhidkovskaya, Nick Winter, Miguel Orbegozo Rodriguez, Robert Lauff, Dustin Wehr, Colin Tang, Zaki Hossain, Shaun Phillips, Fortuna Samuele, Fredrik Ekström, Angela Hammon, Oam Patel, Faraz Farhidi, George Medley, Forough Mohammadzadeh, Madellene Peñaflor, Haile Kassahun, Alena Friedrich, Rayner Hernandez Perez, Daniel Pyda, Taom Sakal, Omkar Dhamane, Ali Khajegili Mirabadi, Eric Hallman, Kenchi Okutsu, Mike Battaglia, Mohammad Maghsoudimehrabani, Alon Amit, Dave Hulbert, Roberto Pereira, Simon Weber, Handoko, Anton Peristyy, Stephen Malina, Mustafa Mehkary, Rami Aly, Frank Reidegeld, Anna-Katharina Dick, Cary Friday, Mukhwinder Singh, Hassan Shapourian, Wanyoung Kim, Mariana Costa, Hubeyb Gurdogan, Harsh Kumar, Chiara Ceconello, Chao Zhuang, Haon Park, Micah Carroll, Andrew R. Tawfeek, Stefan Steinerberger, Daattavya Aggarwal, Michael Kirchhof, Linjie Dai, Evan Kim, Johan Ferret, Jainam Shah, Yuzhou Wang, Minghao Yan, Krzysztof Burdzy, Lixin Zhang, Antonio Franca, Diana T. Pham, Kang Yong Loh, Joshua Robinson, Abram Jackson, Paolo Giordano, Philipp Petersen, Adrian Cosma, Jesus Colino, Colin White, Jacob Votava, Vladimir Vinnikov, Ethan Delaney, Petr Spelda, Vit Stritecky, Syed M. Shahid, Jean-Christophe Mourrat, Lavr Vetoshkin, Koen Sponselee, Renas Bacho, Zheng-Xin Yong, Florencia de la Rosa, Nathan Cho, Xiuyu Li, Guillaume Malod, Orion Weller, Guglielmo Albani, Leon Lang, Julien Laurendeau, Dmitry Kazakov, Fatimah Adesanya, Julien Portier, Lawrence Hollom, Victor Souza, Yuchen Anna Zhou, Julien Degorre, Yiğit Yaln, Gbenga Daniel Obikoya, Rai Michael Pokorny, Filippo Bigi, M. C. Boscá, Oleg Shumar, Kaniuar Bacho, Gabriel Recchia, Mara Popescu, Nikita Shulga, Ngefor Mildred Tanwie, Thomas C. H. Lux, Ben Rank, Colin Ni, Matthew Brooks, Alesia Yakimchyk, Huanxu Quinn Liu, Stefano Cavalleri, Olle Häggström, Emil Verkama, Joshua Newbould, Hans Gundlach, Leonor Brito-Santana, Brian Amaro, Vivek Vajipey, Rynaa Grover, Ting Wang, Yosi Kratish, Wen-Ding Li, Sivakanth Gopi, Andrea Caciolai, Christian Schroeder de Witt, Pablo Hernández-Cámara, Emanuele Rodolà, Jules Robins, Dominic Williamson, Brad Raynor, Hao Qi, Ben Segev, Jingxuan Fan, Sarah Martinson, Erik Y. Wang, Kaylie Hausknecht, Michael P. Brenner, Mao Mao, Christoph Demian, Peyman Kassani, Xinyu Zhang, David Avagian, Eshawn Jessica Scipio, Alon Ragoler, Justin Tan, Blake Sims, Rebeka Plecnik, Aaron Kirtland, Omer Faruk Bodur, D. P. Shinde, Yan Carlos Leyva Labrador, Zahra Adoul, Mohamed Zekry, Ali Karakoc, Tania C. B. Santos, Samir Shamseldeen, Loukmane Karim, Anna Liakhovitskaia, Nate Resman, Nicholas Farina, Juan Carlos Gonzalez, Gabe Maayan, Earth Anderson, Rodrigo De Oliveira Pena, Elizabeth Kelley, Hodjat Mariji, Rasoul Pouriamanesh, Wentao Wu, Ross Finocchio, Ismail Alarab, Joshua Cole, Danyelle Ferreira, Bryan Johnson, Mohammad Safdari, Liangti Dai, Siriphan Arthornthurasuk, Isaac C. McAlister, Alejandro José Moyano, Alexey Pronin, Jing Fan, Angel Ramirez-Trinidad, Yana Malysheva, Daphiny Pottmaier, Omid Taheri, Stanley Stepanic, Samuel Perry, Luke Askew, Raúl Adrián Huerta Rodrguez, Ali M. R. Minissi, Ricardo Lorena, Krishnamurthy Iyer, Arshad Anil Fasiludeen, Ronald Clark, Josh Ducey, Matheus Piza, Maja Somrak, Eric Vergo, Juehang Qin, Benjámin Borbás, Eric Chu, Jack Lindsey, Antoine Jallon, I. M. J. McInnis, Evan Chen, Avi Semler, Luk Gloor, Tej Shah, Marc Carauleanu, Pascal Lauer, Tran Duc Huy, Hossein Shahrtash, Emilien Duc, Lukas Lewark, Assaf Brown, Samuel Albanie, Brian Weber, Warren S. Vaz, Pierre Clavier, Yiyang Fan, Gabriel Poesia Reis e Silva, Long Tony Lian, Marcus Abramovitch, Xi Jiang, Sandra Mendoza, Murat Islam, Juan Gonzalez, Vasilios Mavroudis,

Justin Xu, Pawan Kumar, Laxman Prasad Goswami, Daniel Bugas, Nasser Heydari, Ferenc Jeanplong, Thorben Jansen, Antonella Pinto, Archimedes Apronti, Abdallah Galal, Ng Ze-An, Ankit Singh, Tong Jiang, Joan of Arc Xavier, Kanu Priya Agarwal, Mohammed Berkani, Gang Zhang, Zhehang Du, Benedito Alves de Oliveira Junior, Dmitry Malishev, Nicolas Remy, Taylor D. Hartman, Tim Tarver, Stephen Mensah, Gautier Abou Loume, Wiktor Morak, Farzad Habibi, Sarah Hoback, Will Cai, Javier Gimenez, Roselynn Grace Montecillo, Jakub Łucki, Russell Campbell, Asankhaya Sharma, Khalida Meer, Shreen Gul, Daniel Espinosa Gonzalez, Xavier Alapont, Alex Hoover, Gunjan Chhablani, Freddie Vargus, Arunim Agarwal, Yibo Jiang, Deepakkumar Patil, David Outevsky, Kevin Joseph Scaria, Rajat Maheshwari, Abdelkader Dendane, Priti Shukla, Ashley Cartwright, Sergei Bogdanov, Niels Mündler, Sören Möller, Luca Arnaboldi, Kunvar Thaman, Muhammad Rehan Siddiqi, Prajvi Saxena, Himanshu Gupta, Tony Fruhauff, Glen Sherman, Mátyás Vincze, Siranut Usawasutsakorn, Dylan Ler, Anil Radhakrishnan, Innocent Enyekwe, Sk Md Salauddin, Jiang Muzhen, Aleksandr Maksapetyan, Vivien Rossbach, Chris Harjadi, Mohsen Bahaloohoreh, Claire Sparrow, Jasdeep Sidhu, Sam Ali, Song Bian, John Lai, Eric Singer, Justine Leon Uro, Greg Bateman, Mohamed Sayed, Ahmed Menshawy, Darling Duclosel, Dario Bezzi, Yashaswini Jain, Ashley Aaron, Murat Tiryakioglu, Sheeshram Siddh, Keith Krenek, Imad Ali Shah, Jun Jin, Scott Creighton, Denis Peskoff, Zienab EL-Wasif, Ragavendran P, Michael Richmond, Joseph McGowan, Tejal Patwardhan, Hao-Yu Sun, Ting Sun, Nikola Zubić, Samuele Sala, Stephen Ebert, Jean Kaddour, Manuel Schottdorf, Dianzhuo Wang, Gerol Petruzella, Alex Meiburg, Tilen Medved, Ali ElSheikh, S. Ashwin Hebbar, Lorenzo Vaquero, Xianjun Yang, Jason Poulos, Vilém Zouhar, Sergey Bogdanik, Mingfang Zhang, Jorge Sanz-Ros, David Anugraha, Yinwei Dai, Anh N. Nhu, Xue Wang, Ali Anil Demircali, Zhibai Jia, Yuyin Zhou, Juncheng Wu, Mike He, Nitin Chandok, Aarush Sinha, Gaoxiang Luo, Long Le, Mickaël Noyé, Michał Perełkiewicz, Ioannis Pantidis, Tianbo Qi, Soham Sachin Purohit, Letitia Parcalabescu, Thai-Hoa Nguyen, Genta Indra Winata, Edoardo M. Ponti, Hanchen Li, Kaustubh Dhole, Jongee Park, Dario Abbondanza, Yuanli Wang, Anupam Nayak, Diogo M. Caetano, Antonio A. W. L. Wong, Maria del Rio-Chanona, Dániel Kondor, Pieter Francois, Ed Chalstrey, Jakob Zsambok, Dan Hoyer, Jenny Reddish, Jakob Hauser, Francisco-Javier Rodrigo-Ginés, Suchandra Datta, Maxwell Shepherd, Thom Kamphuis, Qizheng Zhang, Hyunjun Kim, Ruiji Sun, Jianzhu Yao, Franck Dernoncourt, Satyapriya Krishna, Sina Rismanchian, Bonan Pu, Francesco Pinto, Yingheng Wang, Kumar Shridhar, Kalon J. Overholt, Glib Briia, Hieu Nguyen, David Quod Soler Bartomeu, Tony CY Pang, Adam Wecker, Yifan Xiong, Fanfei Li, Lukas S. Huber, Joshua Jaeger, Romano De Maddalena, Xing Han Lù, Yuhui Zhang, Claas Beger, Patrick Tser Jern Kon, Sean Li, Vivek Sanker, Ming Yin, Yihao Liang, Xinlu Zhang, Ankit Agrawal, Li S. Yifei, Zechen Zhang, Mu Cai, Yasin Sonmez, Costin Cozianu, Changhao Li, Alex Slen, Shoubin Yu, Hyun Kyu Park, Gabriele Sarti, Marcin Briański, Alessandro Stolfo, Truong An Nguyen, Mike Zhang, Yotam Perlitz, Jose Hernandez-Orallo, Runjia Li, Amin Shabani, Felix Juefei-Xu, Shikhar Dhingra, Orr Zohar, My Chiffon Nguyen, Alexander Pondaven, Abdurrahim Yilmaz, Xuandong Zhao, Chuanyang Jin, Muyan Jiang, Stefan Todoran, Xinyao Han, Jules Kreuer, Brian Rabern, Anna Plassart, Martino Maggetti, Luther Yap, Robert Geirhos, Jonathon Kean, Dingsu Wang, Sina Mollaei, Chenkai Sun, Yifan Yin, Shiqi Wang, Rui Li, Yaowen Chang, Anjiang Wei, Alice Bizeul, Xiaohan Wang, Alexandre Oliveira Arrais, Kushin Mukherjee, Jorge Chamorro-Padial, Jiachen Liu, Xingyu Qu, Junyi Guan, Adam Bouyamourn, Shuyu Wu, Martyna Plomecka, Junda Chen, Mengze Tang, Jiaqi Deng, Shreyas Subramanian, Haocheng Xi, Haoxuan Chen, Weizhi Zhang, Yinuo Ren, Haoqin Tu, Sejong Kim, Yushun Chen, Sara Vera Marjanović, Junwoo Ha, Grzegorz Luczyna, Jeff J. Ma, Zewen Shen, Dawn Song, Cedegao E. Zhang, Zhun Wang, Gaël Gendron, Yunze Xiao, Leo Smucker, Erica Weng, Kwok Hao Lee, Zhe Ye, Stefano Ermon, Ignacio D. Lopez-Miguel, Theo Knights, Anthony Gitter, Namkyu Park, Boyi Wei, Hongzheng Chen, Kunal Pai, Ahmed Elkhanany, Han Lin, Philipp D. Siedler, Jichao Fang, Ritwik Mishra, Károly Zsolnai-Fehér, Xilin Jiang, Shadab Khan, Jun Yuan, Rishab Kumar Jain, Xi Lin, Mike Peterson, Zhe Wang, Aditya Malusare, Maosen Tang, Isha Gupta, Ivan Fosin, Timothy Kang, Barbara Dworakowska, Kazuki Matsumoto, Guangyao Zheng, Gerben Sewuster, Jorge Pretel Villanueva, Ivan Rannev, Igor Chernyavsky, Jiale Chen, Deepayan Banik, Ben Racz, Wenchao Dong, Jianxin Wang, Laila Bashmal, Duarte V. Gonçalves, Wei Hu, Kaushik Bar, Ondrej Bohdal, Atharv Singh Patlan, Shehzaad Dhuliawala, Caroline Geirhos, Julien Wist, Yuval Kansal, Bingsen Chen, Kutay Tire, Atak Talay Yücel, Brandon Christof, Veerupaksh Singla, Zijian Song, Sanxing Chen, Jiaxin Ge, Kaustubh Ponkshe, Isaac Park, Tianneng Shi, Martin Q. Ma, Joshua Mak, Sherwin Lai, Antoine Moulin, Zhuo Cheng, Zhanda Zhu, Ziyi Zhang, Vaidehi Patil, Ketan Jha, Qiutong Men, Jiaxuan Wu, Tianchi Zhang, Bruno Hebling Vieira, Alham Fikri Aji, Jae-Won Chung, Mohammed

Mahfoud, Ha Thi Hoang, Marc Sperzel, Wei Hao, Kristof Meding, Sihan Xu, Vassilis Kostakos, Davide Manini, Yueying Liu, Christopher Toukmaji, Eunmi Yu, Arif Engin Demircali, Zhiyi Sun, Ivan Dewerpe, Hongsen Qin, Roman Pflugfelder, James Bailey, Johnathan Morris, Ville Heilala, Sybille Rosset, Zishun Yu, Peter E. Chen, Woongyeong Yeo, Eeshaan Jain, Sreekar Chigurupati, Julia Chernyavsky, Sai Prajwal Reddy, Subhashini Venugopalan, Hunar Batra, Core Francisco Park, Hieu Tran, Guilherme Maximiano, Genghan Zhang, Yizhuo Liang, Hu Shiyu, Rongwu Xu, Rui Pan, Siddharth Suresh, Ziqi Liu, Samaksh Gulati, Songyang Zhang, Peter Turchin, Christopher W. Bartlett, Christopher R. Scotese, Phuong M. Cao, Ben Wu, Jacek Karwowski, and Davide Scaramuzza. A benchmark of expert-level academic questions to assess ai capabilities. *Nature*, 649(8099):1139–1146, January 2026. ISSN 1476-4687. doi: 10.1038/s41586-025-09962-4. URL http://dx.doi.org/10.1038/s41586-025-09962-4.

Clark Barrett, Pascal Fontaine, and Cesare Tinelli. The satisfiability modulo theories library (SMT-LIB). www.SMT-LIB.org, 2016.

Jifan Chen, Eunsol Choi, and Greg Durrett. Can NLI models verify QA systems' predictions? In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2021*, pp. 3841–3854, Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.findings-emnlp.324. URL https://aclanthology.org/2021.findings-emnlp.324/.

Bhavana Dalvi, Peter Jansen, Oyvind Tafjord, Zhengnan Xie, Hannah Smith, Leighanna Pipatanangkura, and Peter Clark. Explaining answers with entailment trees. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih (eds.), *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 7358–7370, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.585. URL https://aclanthology.org/2021.emnlp-main.585/.

Leonardo De Moura and Nikolaj Bjørner. Z3: an efficient smt solver. In *Proceedings of the Theory and Practice of Software, 14th International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, TACAS'08/ETAPS'08, pp. 337–340, Berlin, Heidelberg, 2008. Springer-Verlag. ISBN 3540787992.

William Fedus, Barret Zoph, and Noam Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120):1–39, 2022. URL https://jmlr.org.

Yu Feng, Nathaniel Weir, Kaj Bostrom, Sam Bayless, Darion Cassel, Sapana Chaudhary, Benjamin Kiesl-Reiter, and Huzefa Rangwala. Vericot: Neuro-symbolic chain-of-thought validation via logical consistency checks, 2025. URL https://arxiv.org/abs/2511.04662.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Hanwei Xu, Honghui Ding, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jingchang Chen, Jingyang Yuan, Jinhao Tu, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaichao You, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingxu Zhou, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu

Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638, 2025. ISSN 1476-4687. doi: 10.1038/s41586-025-09422-z. URL http://dx.doi.org/10.1038/s41586-025-09422-z.

Simeng Han, Hailey Schoelkopf, Yilun Zhao, Zhenting Qi, Martin Riddell, Wenfei Zhou, James Coady, David Peng, Yujie Qiao, Luke Benson, Lucy Sun, Alexander Wardle-Solano, Hannah Szabó, Ekaterina Zubova, Matthew Burtell, Jonathan Fan, Yixin Liu, Brian Wong, Malcolm Sailor, Ansong Ni, Linyong Nan, Jungo Kasai, Tao Yu, Rui Zhang, Alexander Fabbri, Wojciech Maciej Kryscinski, Semih Yavuz, Ye Liu, Xi Victoria Lin, Shafiq Joty, Yingbo Zhou, Caiming Xiong, Rex Ying, Arman Cohan, and Dragomir Radev. FOLIO: Natural language reasoning with first-order logic. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 22017–22031, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.1229. URL https://aclanthology.org/2024.emnlp-main.1229/.

Alon Jacovi, Yonatan Bitton, Bernd Bohnet, Jonathan Herzig, Or Honovich, Michael Tseng, Michael Collins, Roee Aharoni, and Mor Geva. A chain-of-thought is as strong as its weakest link: A benchmark for verifiers of reasoning chains. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 4615–4634, Bangkok, Thailand, 2024a.

Alon Jacovi, Yonatan Bitton, Bernd Bohnet, Jonathan Herzig, Or Honovich, Michael Tseng, Michael Collins, Roee Aharoni, and Mor Geva. A chain-of-thought is as strong as its weakest link: A benchmark for verifiers of reasoning chains, 2024b. URL https://arxiv.org/abs/2402.00559.

Wouter Kool, Herke van Hoof, and Max Welling. Buy 4 reinforce samples, get a baseline for free! In *DeepRLStructPred@ICLR*, 2019. URL https://api.semanticscholar.org/CorpusID:198489118.

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, 2023. URL https://arxiv.org/abs/2305.20050.

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 *The Twelfth International Conference on Learning Representations*, 2024.

Stephanie Lin, Jacob Hilton, and Owain Evans. Truthfulqa: Measuring how models mimic human falsehoods, 2022. URL https://arxiv.org/abs/2109.07958.

Zhan Ling, Yunhao Fang, Xuanlin Li, Zhining Shi, Hoifung Lee, Minjie Su, and Ran Xu. Deductive verification of chain-of-thought reasoning. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

Jian Liu, Leyang Cui, Hanmeng Liu, Dandan Huang, Yile Wang, and Yue Zhang. Logiqa: A challenge dataset for machine reading comprehension with logical reasoning, 2020. URL https://arxiv.org/abs/2007.08124.

Sagnik Mukherjee, Abhinav Chinta, Takyoung Kim, Tarun Anoop Sharma, and Dilek Hakkani-Tür. Premise-augmented reasoning chains improve error identification in math reasoning with llms, 2025. URL https://arxiv.org/abs/2502.02362.

Theo X. Olausson, Eirini Platanios, Roozbeh Mottaghi, and Jacob Andreas. LINC: A neurosymbolic approach for logical reasoning by combining language models with first-order logic provers. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, 2023.

OpenAI, :, Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, Alex Iftimie, Alex Karpenko, Alex Tachard Passos, Alexander Neitz, Alexander Prokofiev, Alexander Wei, Allison Tam, Ally Bennett, Ananya Kumar, Andre Saraiva, Andrea Vallone, Andrew Duberstein, Andrew Kondrich, Andrey Mishchenko, Andy Applebaum, Angela Jiang, Ashvin Nair, Barret Zoph, Behrooz Ghorbani, Bohan Zhang, Ben Rossen, Benjamin Sokolowsky, Boaz Barak, Bob McGrew, Borys Minaiev, Botao Hao, Bowen Baker, Brandon Houghton, Brandon McKinzie, Brydon Eastman, Camillo Lugaresi, Cary Bassin, Cary Hudson, Chak Ming Li, Charles de Bourcy, Chelsea Voss, Chen Shen, Chong Zhang, Chris Koch, Chris Orsinger, Christopher Hesse, Claudia Fischer, Clive Chan, Dan Roberts, Daniel Kappler, Daniel Levy, Daniel Selsam, David Dohan, David Farhi, David Mely, David Robinson, Dimitris Tsipras, Doug Li, Dragos Oprica, Eben Freeman, Eddie Zhang, Edmund Wong, Elizabeth Proehl, Enoch Cheung, Eric Mitchell, Eric Wallace, Erik Ritter, Evan Mays, Fan Wang, Felipe Petroski Such, Filippo Raso, Florencia Leoni, Foivos Tsimpourlas, Francis Song, Fred von Lohmann, Freddie Sulit, Geoff Salmon, Giambattista Parascandolo, Gildas Chabot, Grace Zhao, Greg Brockman, Guillaume Leclerc, Hadi Salman, Haiming Bao, Hao Sheng, Hart Andrin, Hessam Bagherinezhad, Hongyu Ren, Hunter Lightman, Hyung Won Chung, Ian Kivlichan, Ian O'Connell, Ian Osband, Ignasi Clavera Gilaberte, Ilge Akkaya, Ilya Kostrikov, Ilya Sutskever, Irina Kofman, Jakub Pachocki, James Lennon, Jason Wei, Jean Harb, Jerry Twore, Jiacheng Feng, Jiahui Yu, Jiayi Weng, Jie Tang, Jieqi Yu, Joaquin Quiñonero Candela, Joe Palermo, Joel Parish, Johannes Heidecke, John Hallman, John Rizzo, Jonathan Gordon, Jonathan Uesato, Jonathan Ward, Joost Huizinga, Julie Wang, Kai Chen, Kai Xiao, Karan Singhal, Karina Nguyen, Karl Cobbe, Katy Shi, Kayla Wood, Kendra Rimbach, Keren Gu-Lemberg, Kevin Liu, Kevin Lu, Kevin Stone, Kevin Yu, Lama Ahmad, Lauren Yang, Leo Liu, Leon Maksin, Leyton Ho, Liam Fedus, Lilian Weng, Linden Li, Lindsay McCallum, Lindsey Held, Lorenz Kuhn, Lukas Kondraciuk, Lukasz Kaiser, Luke Metz, Madelaine Boyd, Maja Trebacz, Manas Joglekar, Mark Chen, Marko Tintor, Mason Meyer, Matt Jones, Matt Kaufer, Max Schwarzer, Meghan Shah, Mehmet Yatbaz, Melody Y. Guan, Mengyuan Xu, Mengyuan Yan, Mia Glaese, Mianna Chen, Michael Lampe, Michael Malek, Michele Wang, Michelle Fradin, Mike McClay, Mikhail Pavlov, Miles Wang, Mingxuan Wang, Mira Murati, Mo Bavarian, Mostafa Rohaninejad, Nat McAleese, Neil Chowdhury, Neil Chowdhury, Nick Ryder, Nikolas Tezak, Noam Brown, Ofir Nachum, Oleg Boiko, Oleg Murk, Olivia Watkins, Patrick Chao, Paul Ashbourne, Pavel Izmailov, Peter Zhokhov, Rachel Dias, Rahul Arora, Randall Lin, Rapha Gontijo Lopes, Raz Gaon, Reah Miyara, Reimar Leike, Renny Hwang, Rhythm Garg, Robin Brown, Roshan James, Rui Shu, Ryan Cheu, Ryan Greene, Saachi Jain, Sam Altman, Sam Toizer, Sam Toyer, Samuel Miserendino, Sandhini Agarwal, Santiago Hernandez, Sasha Baker, Scott McKinney, Scottie Yan, Shengjia Zhao, Shengli Hu, Shibani Santurkar, Shraman Ray Chaudhuri, Shuyuan Zhang, Siyuan Fu, Spencer Papay, Steph Lin, Suchir Balaji, Suvansh Sanjeev, Szymon Sidor, Tal Broda, Aidan Clark, Tao Wang, Taylor Gordon, Ted Sanders, Tejal Patwardhan, Thibault Sottiaux, Thomas Degry, Thomas Dimson, Tianhao Zheng, Timur Garipov, Tom Stasi, Trapit Bansal, Trevor Creech, Troy Peterson, Tyna Eloundou, Valerie Qi, Vineet Kosaraju, Vinnie Monaco, Vitchyr Pong, Vlad Fomenko, Weiyi Zheng, Wenda Zhou, Wenting Zhan, Wes McCabe, Wojciech Zaremba, Yann Dubois, Yinghai Lu, Yining Chen, Young Cha, Yu Bai, Yuchen He, Yuchen Zhang, Yunyun Wang, Zheng Shao, and Zhuohan Li. Openai o1 system card, 2026. URL https://arxiv.org/abs/2412.16720.

Liangming Pan, Alon Albalak, Xinyi Wang, and William Wang. Logic-LM: Empowering large language models with symbolic solvers for faithful logical reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, 2023a.

Liangming Pan, Alon Albalak, Xinyi Wang, and William Yang Wang. Logic-LM: empowering large language models with symbolic solvers for faithful logical reasoning. In *Findings of the 2023 Conference on Empirical Methods in Natural Language Processing (Findings of EMNLP)*, Singapore, Dec 2023b. URL https://arxiv.org/abs/2305.12295.

Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan (eds.), *Proceedings of*

the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pp. 3982–3992, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1410. URL https://aclanthology.org/D19-1410/.

Zacchary Sadeddine and Fabian M. Suchanek. Verifying the steps of deductive reasoning chains. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 456–475, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-256-5. doi: 10.18653/v1/2025.findings-acl.25. URL https://aclanthology.org/2025.findings-acl.25/.

Abulhair Saparov and He He. Language models are greedy reasoners: A systematic formal analysis of chain-of-thought, 2023. URL https://arxiv.org/abs/2210.01240.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *International Conference on Learning Representations (ICLR)*, 2017. URL https://openreview.net/forum?id=B1ckMDqlg.

Xu Shen, Song Wang, Zhen Tan, Laura Yao, Xinyu Zhao, Kaidi Xu, Xin Wang, and Tianlong Chen. Faithcot-bench: Benchmarking instance-level faithfulness of chain-of-thought reasoning, 2026. URL https://arxiv.org/abs/2510.04040.

Jiasheng Si, Yibo Du, Yingjie Zhang, Wenjuan Zhao, Xuefeng Liu, Libo Sun, and Deyu Zhou. CheckWhy: Causal fact verification via argument structure. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, 2024.

Oyvind Tafjord, Bhavana Dalvi, and Peter Clark. ProofWriter: Generating implications, proofs, and abductive statements over natural language. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli (eds.), *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pp. 3621–3634, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.findings-acl.317. URL https://aclanthology.org/2021.findings-acl.317/.

Miles Turpin, Julian Michael, Ethan Perez, and Samuel R. Bowman. Language models don't always say what they think: Unfaithful explanations in chain-of-thought prompting. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

Yuhuai Wu, Albert Q. Jiang, Wenda Li, Markus N. Rabe, Charles Staats, Mateja Jamnik, and Christian Szegedy. Autoformalization with large language models, 2022. URL https://arxiv.org/abs/2205.12615.

A Routing Framework

Figure 1 illustrates the routing framework at the inference time. It consists of the following components:

- **Router:** Given a verification instance as input, the router decides whether to select the *Formal* verifier/route/tool or the *Informal* verifier/route/tool for that instance. We also experimented with a *Hybrid* verifier/route, which determines the final prediction by taking a majority vote over the outputs of the formal and informal verifiers after running each of them (K) times. However, this route was selected only rarely by the router. The routing system prompt is shown in Figure 8. We use `Qwen-2.5-7B-Instruct` as the routing model.

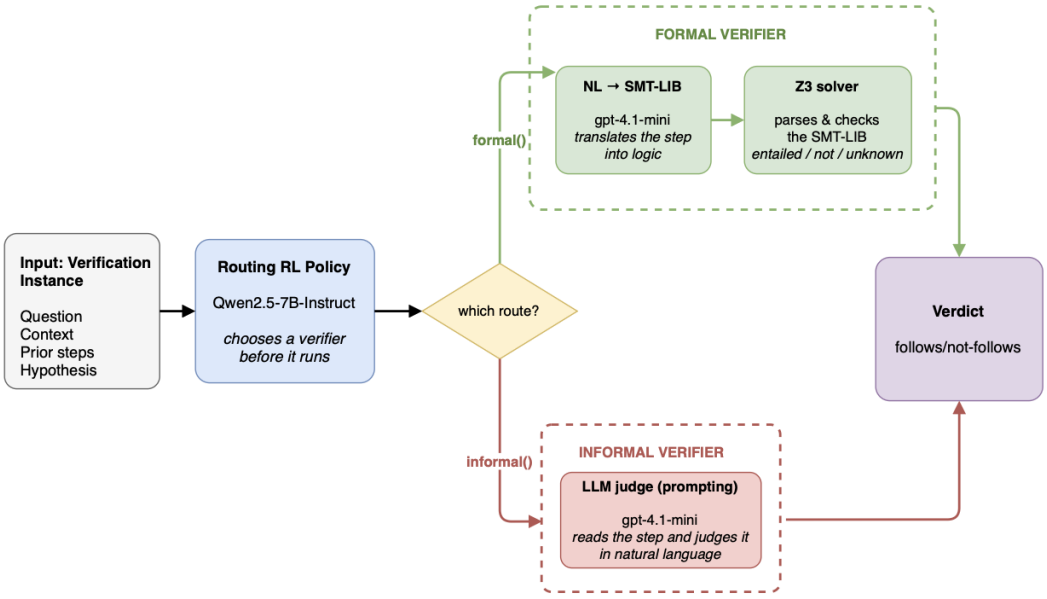

Figure 1: Routing Framework Overview: Inference time.

Table 2: Macro-F1 score (%) on ProofWriter dataset. Here, **Verifier LLM** represents the model used as both Formal and Informal Verification tool, while **Method** represents the different baselines, and approaches. Our proposed RL Policy approach uses `Qwen-2.5-7B-Instruct` for routing. **Bold** and underline mark the best and second-best scores, respectively, among the GPT-4.1-mini and GPT-OSS-20B rows; GPT-5.4 and GPT-4.1 are reported as reference (near-upper-bound) baselines and are excluded from this comparison. A dash (–) marks an (LLM, method) pair not evaluated on that dataset.

Verifier LLM	Method	ProofWriter		
		Neg	Hallu	Remove
GPT-5.4	Informal Only	100.00	89.74	91.33
GPT-4.1	VANESSA	88.83	84.93	86.31
	PARC-GRAPH	58.65	56.51	62.52
	Informal Only	97.61	92.04	98.14
GPT-4.1-mini	Formal Only	72.63	64.54	63.06
	Informal Only	90.71	85.67	92.28
	RL Policy	**98.40**	**86.21**	93.09
GPT-OSS-20B	Formal Only	73.33	64.46	65.49
	RL Policy	87.31	84.58	**95.22**

- **Formal Verifier/Route/Tool:** When selected, the formal verifier first translates the verification instance from natural language into the SMT-LIB v2.6 format using the prompt shown in Figure 2. The resulting SMT-LIB program is then executed using the Z3 solver to produce the final prediction, which is mapped to the labels `follows` or `not-follows`. We use `gpt-4.1-mini` to perform the translation from natural language to SMT-LIB.
- **Informal Verifier/Route/Tool:** When selected, the informal verifier directly predicts the labels (`follows`/ `not-follows`) using an LLM and the prompt shown in Figure 3. We used various closed source model for the informal route like `gpt-4.1-mini`, and `gpt-4.1`.

A.1 Result on ProofWriter Dataset

B Prompts

This section shows the prompts used in our experiments for different tasks like: 1) Prompt for translating natural language instances to SMT-LIB v2.6, 2) Prompt to verify using the informal tool, 3) Prompt used in calculating the Z_F score, 4) Prompt used in calculating the Z_C score, 5) Prompt used by the RL routing policy to select one of the verification routes for a reasoning step, and 6) Prompt used to generating explanation for a given gold route.

B.1 Translate Natural Language Instances to SMT-LIB v2.6

Figure 2 shows the prompt for translating natural language instances to SMT-LIB v2.6

```
You are an expert at translating natural language into SMT-LIB v2.6
for the Z3 theorem prover.

You may be given the following four blocks of natural language:

QUESTION:
The question being answered.  Use it only to interpret the vocabulary.
Do not translate or assert it unless it also appears in the context or
prior steps.

CONTEXT:
Background facts and rules.

PRIOR STEPS:
The reasoning completed so far.

HYPOTHESIS:
The single reasoning step to be verified.

The QUESTION, CONTEXT, or PRIOR STEPS may be absent.
Translate the provided CONTEXT (if any), PRIOR STEPS (if any), and
HYPOTHESIS into a single shared SMT-LIB representation.

Guidelines
 • Output only valid SMT-LIB v2.6 syntax.
 • Use a single shared vocabulary throughout the translation.  If two
   expressions refer to the same entity, property, or relation, they
   must be represented by the same SMT symbol.
 • Translate only the information explicitly stated in the input.  Do
   not invent facts, assumptions, or modelling choices.
 • If any part of the input cannot be faithfully represented in
   classical SMT, record the original text span in untranslatable and
   omit it from the formalization.  Do not approximate untranslatable
   statements.
 • If the hypothesis depends on untranslatable content, set goal to
   "".

Output
Output only valid JSON with exactly the following keys.  Do not
include Markdown, comments, or trailing commas.

{
  "declarations":
      "<one JSON string containing all declaration
       commands separated by \n>",

  "premises": [
      "<one complete (assert ...) command for each
       statement in the CONTEXT or PRIOR STEPS>"
  ],

  "goal":
      "<a single SMT-LIB Boolean term for the
       hypothesis; do not wrap it in (assert ...);
       use "" if untranslatable>",

  "untranslatable": [
      "<verbatim text span>", ...
  ],

  "notes":
      "<brief explanation describing important
       modelling decisions>"
```

```
}
```

Figure 2: Prompt used to translate natural language statements into SMT-LIB v2.6 for theorem proving with Z3.

B.2 Prompt to verify a given reasoning instance using the informal tool.

Figure 3 shows the prompt to verify a given reasoning instance using the informal tool.

```
You are an expert in logical reasoning.

Your task is to determine whether the HYPOTHESIS logically follows
from the information provided.

You may be given a QUESTION, CONTEXT, PRIOR STEPS, and a HYPOTHESIS.
The QUESTION, CONTEXT, or PRIOR STEPS may be absent.

Instructions
 • Treat all provided information as true.
 • Use only the information provided in the input.
 • Do not rely on outside knowledge or make additional assumptions.
 • The hypothesis is supported only if it can be logically inferred
   from the provided information.
 • If the information is insufficient to conclude the hypothesis, or
   if the hypothesis contradicts any earlier information, it is not a
   valid next step.

Judge whether the hypothesis is justified at this point in the
reasoning process, not whether it happens to be true in general.

Think briefly, then commit to a binary verdict.

Output
Return STRICT JSON:

{
  "label": 0 or 1,
  "confidence": <float 0..1>,
  "explanation": "<Brief explanation of the label>"
}

label = 1 means the hypothesis is a valid next step.
label = 0 means it is not.
```

Figure 3: Prompt used to verify whether a hypothesis is a logically valid next reasoning step given the available question, context, and prior reasoning.

B.3 Z_F score prompt

Figure 4, 7, and 8 contains the complete prompt used in calculating the Z_F score.

```
You are an expert in formal knowledge representation.
Your task is to determine whether a TEXT PASSAGE (one or more
sentences, to be formalised together as a single unit) can be
faithfully represented in SMT-LIB using the expressive power supported
by Z3 (many-sorted first-order logic together with built-in theories
such as integers, reals, arithmetic, equality, and ordering).
Do NOT translate the passage into SMT-LIB.
Only analyze its semantic content.
Judge the passage AS A WHOLE: report every phenomenon that appears
anywhere in it.

Step 1.  Identify Semantic Phenomena
```

A passage may contain multiple phenomena, possibly across different sentences. Refer to each phenomenon by its NAME, exactly as written below.

Expressible phenomena

binary_predicates
Relations or properties of entities.

universal_quantification
Statements about all members of a class.

existential_quantification
Statements asserting the existence of one or more entities.

negation Explicit logical negation.

arithmetic_or_counting
Numbers, counting, arithmetic, measurements, ratios, percentages, or explicit numeric thresholds.

ordering_or_temporal
Temporal ordering or ordering over numeric or ordinal domains.

Not faithfully expressible

modal_meaning Necessity, possibility, obligation, permission, or similar modal notions.

generic_or_default
Generalizations that implicitly admit exceptions.

gradable_without_threshold
Context-dependent adjectives or comparisons whose truth depends on an implicit standard.

IMPORTANT If the passage contains an explicit numeric threshold, classify it as arithmetic_or_counting instead.

counterfactual_conditional
Hypothetical situations contrary to known facts.

causal_claim Statements whose meaning depends on causal relationships rather than logical implication.

performative Utterances that perform an action simply by being spoken.

Figure 4: Prompt used to determine whether a given passage can be faithfully represented in SMT-LIB (Part 1).

Step 2. Determine Centrality
Only for the NOT-faithfully-expressible phenomena, determine whether the phenomenon is:

- central: The phenomenon is essential to the main meaning or truth conditions of the passage.
  For a **MULTI-SENTENCE** passage, central means the passage cannot be faithfully formalised without it -- e.g., a rule the rest of the reasoning depends on is modal or causal.
- peripheral: The phenomenon is present, but removing it would preserve most of the passage's logical content -- e.g., it appears in one incidental sentence while the rest of the passage formalises cleanly.

The expressible phenomena are fully expressible and therefore do **NOT** require a centrality field.

Output

Return **STRICT JSON** in exactly this schema:

```
  "phenomena": [
      "phenomenon": "<name>",
      "phenomenon": "<name>",
       "centrality": "<central|peripheral>"
  ],
  "reason": "<one short sentence>"
```

Rules for the output

- "phenomenon" MUST be one of the twelve NAMES listed above, copied exactly. It is a PHENOMENON NAME, never a sentence number, index, or position in the passage.
- Include one entry per DISTINCT phenomenon present. Do NOT emit one entry per sentence: a passage of 20 sentences that only uses universal quantification and negation has exactly TWO entries.
- Include "centrality" if and only if the phenomenon is one of the NOT-faithfully-expressible ones.
- List only phenomena that are actually present. There is no fixed number of entries.

Figure 5: Prompt used to determine whether a given passage can be faithfully represented in SMT-LIB (Part 2 continued).

Examples

Passage
Every student passed because John cheated.

Output

```
  "phenomena":
  [
      "phenomenon": "universal_quantification",
      "phenomenon": "causal_claim",
       "centrality": "peripheral"
  ],
  "reason": "The main claim is universal quantification; the causal
             statement is secondary."
```

Passage
Every dog is a mammal. Rex is a dog. Rex is not a cat. Birds generally fly.

Output

```
  "phenomena":
  [
      "phenomenon": "universal_quantification",
      "phenomenon": "binary_predicates",
      "phenomenon": "negation",
      "phenomenon": "generic_or_default",
       "centrality": "peripheral"
  ],
  "reason": "The passage is mostly clean quantification and negation;
             only the last sentence is a defeasible generic."
```

Figure 6: Prompt used to determine whether a given passage can be faithfully represented in SMT-LIB (Part 3 continued).

B.4 Z_C SCORE PROMPT

Figure 7 contains prompt used to translate SMT-LIB formulas into semantically faithful English sentences (used for calculating Z_C score).

```
You are an SMT-LIB-to-English translator.
You are given:

    1. The declarations of an SMT-LIB program, which define the
       meaning of predicate, function, constant, and sort names.
    2. ONE OR MORE SMT-LIB formulas, one per line.  Treat them as a
       CONJUNCTION: the passage you produce must assert all of them.

Your task is to convert the SMT-LIB formulas into English text whose
meaning is as close as possible to the formulas.  Semantic fidelity is
much more important than stylistic naturalness.

Rules

     • Use the declarations only to interpret the symbols.  Do not
       infer or invent any additional meaning.
     • Translate exactly what the formula states.  Do not add, omit,
       weaken, strengthen, or paraphrase its meaning.
     • Preserve every logical component of the formula, including:
         – universal and existential quantification,
         – negation,
         – conjunction and disjunction,
         – implication and equivalence,
         – equality and inequality,
         – arithmetic expressions, comparisons, and numeric values.
     • Whenever possible, express quantified formulas in natural
       English (e.g., ``every...'', ``some...'', ``no...'')  rather
       than referring to logical variables.
     • Prefer a literal rendering of the logical meaning over a
       stylistically smoother paraphrase.
     • Produce ONE declarative English sentence per input formula,
       in the same order, joined into a single passage.  If there is
       exactly one formula, produce exactly one sentence.
     • Do not explain the translation.
     • Do not mention SMT-LIB, Z3, predicates, variables, formulas, or
       logic.

Output
Return STRICT JSON:

  "sentence": "<English text: one sentence per input formula, in order>"
```

Figure 7: Prompt used to translate SMT-LIB formulas into semantically faithful English sentences.

B.5 SELECT THE VERIFICATION ROUTE

Figure 8 shows the prompt used by the RL routing policy to select one of the verification routes for a reasoning step.

You are an expert at selecting the best verification route for a single reasoning step.

You are given a reasoning instance consisting of a Question (if present), a Context (if present), Prior Steps (if present), and a Hypothesis. The goal of the underlying reasoning task is to determine whether the Hypothesis follows from the available information. You are NOT asked to determine that yourself. Your task is to choose the verification route that is most appropriate for verifying this reasoning step.

Verification Routes

Three verification routes are available:

- formal() --- verifies the reasoning by translating it into SMT-LIB and proving it with the Z3 theorem prover. This route is generally most suitable when the required reasoning can be faithfully represented using formal logic, such as symbolic, arithmetic, quantified, counting, ordering, or other precisely defined logical reasoning.
- informal() --- verifies the reasoning using natural-language reasoning. This route is generally most suitable when the reasoning depends on commonsense, world knowledge, lexical semantics, or information that cannot be faithfully formalized.
- hybrid() --- combines evidence from both routes when each contributes useful information and neither route alone is clearly sufficient.

Instructions

First explain your reasoning, then choose exactly one route.

Your explanation must rely only on the characteristics of the reasoning instance itself --- for example, the type of reasoning involved, whether the information in the instance (context/prior steps/hypothesis) can be faithfully formalized, whether commonsense or world knowledge is required, or why combining both styles of reasoning is beneficial.

Do not assume access to the outputs of any verification route. Do not mention SMT translations, Z3 results, confidence scores, or whether any route succeeds or fails.

Write a detailed, instance-specific explanation. Justify your conclusion by discussing multiple relevant characteristics of the reasoning instance. Ground every argument in the given instance, avoid generic descriptions of the routes, and do not simply restate the route definitions.

Output Format

Respond in exactly this format:

<think>...</think><route>ROUTE()</route>

where ROUTE is one of: formal, informal, or hybrid. Do not output anything after </route>.

Figure 8: Prompt used by the router to select the verification route (formal, informal, or hybrid) for a reasoning step.

B.6 Generating explanation for a given gold route

Figure 9 shows the labelling prompt used to generate the blind, instance-specific explanation of the gold route.

Labelling Prompt for Verification Route Selection

You are given a reasoning instance consisting of a **Question** (if present), a **Context** (if present), **Prior Steps** (if present), and a **Hypothesis**.
The goal of the underlying reasoning task is to determine whether the Hypothesis follows from the available information.
Three verification routes are available:

- **FORMAL** -- verifies the reasoning by translating it into SMT-LIB and proving it with the Z3 theorem prover. This route is generally most suitable when the required reasoning can be faithfully represented using formal logic, such as symbolic, arithmetic, quantified, counting, ordering, or other precisely defined logical reasoning.
- **INFORMAL** -- verifies the reasoning using natural-language reasoning. This route is generally most suitable when the reasoning depends on commonsense, world knowledge, lexical semantics, or information that cannot be faithfully formalized.
- **HYBRID** -- combines evidence from both routes when each contributes useful information and neither route alone is clearly sufficient.

For the reasoning instance below, the most appropriate verification route has already been determined to be:
{chosen}
Your task is to explain why this route is the most appropriate choice.
Your explanation should rely only on the characteristics of the reasoning instance itself -- for example, the type of reasoning involved, whether the information in a given reasoning instance (context/prior steps/hypothesis) can be faithfully formalized, whether commonsense or world knowledge is required, or why combining both styles of reasoning is beneficial.
Do not assume access to the outputs of any verification route.
Do not mention SMT translations, Z3 results, confidence scores, or whether any route succeeds or fails.
Write a detailed, instance-specific explanation.
Justify your conclusion by discussing multiple relevant characteristics of the reasoning instance.
Ground every argument in the given instance, avoid generic descriptions of the routes, and do not simply restate the route definitions.

Output
Return only valid JSON in the following format:

```
"reasoning": "<your detailed explanation>"
```

INSTANCE
{instance_text}

Figure 9: The labelling prompt used to generate the blind, instance-specific explanation of the gold route. The placeholders {chosen} and {instance_text} are filled with the gold route phrase and the rendered reasoning instance, respectively.

C Algorithms

C.1 Main algorithm

Algorithm 1 describes how the routing policy is trained in two stages. In the first stage we warm up the policy with supervised fine-tuning: for each instance we build a prompt from its question, context, prior steps, and hypothesis, and train the model to produce the gold route wrapped in its reasoning, using a cross-entropy loss computed over the completion only. Once the warm-up finishes we keep a frozen copy of the policy as a reference model, which later anchors the policy so that

Algorithm 1 Two-stage training of the verification-routing policy

Require: training set $\mathcal{D} = \{(x_i, a_i^\star, y_i)\}$ with input instance x_i, gold route $a^\star$, and label $y_i \in$ {follows/ not-follows}; base LM with LoRA adapter π_θ; group size G; KL weight β; duals μ_F, μ_C; tolerance ϵ; dual step η; clip bound $\mu_{\max}$

Stage 1: Supervised fine-tuning (warm-up)

for each mini-batch $B \subset \mathcal{D}$ **do**
  for each $(x, a^\star) \in B$ **do**
    build prompt from x; target completion $= \langle\texttt{think}\rangle\, \rho\, \langle\texttt{route}\rangle a^\star()$
    $\ell \leftarrow$ token cross-entropy on the completion only (prompt masked)
  end for
  update θ by $\nabla_\theta \operatorname{mean}(\ell)$ ▷ cosine LR schedule with warm-up
end for
$\pi_{\text{ref}} \leftarrow \pi_\theta$ ▷ freeze a copy as the KL anchor

Stage 2: RL fine-tuning (RLOO)

for each epoch **do**
  for each mini-batch $B \subset \mathcal{D}$ **do**
    $\mathcal{Z}_F \leftarrow [\,]$; $\mathcal{Z}_C \leftarrow [\,]$ ▷ collect faithfulness scores for the dual update
    for each instance $x \in B$ **do**
      for $g = 1$ **to** G **do**
        sample route $a_g \sim \pi_\theta(\cdot \mid x)$ using prompt show in figure 8 ▷ one rollout
        $r_g,\, Z_F^g,\, Z_C^g \leftarrow \textsc{StepReward}(x, a_g)$ ▷ Algorithm 2
        if $a_g \in \{\text{FORMAL}, \text{HYBRID}\}$ **then**
          append Z_F^g to $\mathcal{Z}_F$; append Z_C^g to $\mathcal{Z}_C$
        end if
      end for
      $A_g \leftarrow r_g - \frac{1}{G-1}\sum_{j \neq g} r_j$ for all g ▷ RLOO leave-one-out advantage
    end for
    $\mathcal{L}_{\text{PG}} \leftarrow -\dfrac{1}{|B|G}\sum_x \sum_g A_g \log \pi_\theta(a_g \mid x)$ ▷ policy-gradient (REINFORCE) surrogate loss
    $\mathcal{L}_{\text{KL}} \leftarrow \text{mean}\ \mathrm{KL}(\pi_\theta \,\|\, \pi_{\text{ref}})$ ▷ token-level k_3 estimator
    update θ by $\nabla_\theta(\mathcal{L}_{\text{PG}} + \beta\, \mathcal{L}_{\text{KL}})$

Dual (Lagrangian) update on the faithfulness constraints:

    $\bar{m}_F \leftarrow \operatorname{mean}\big(1 - Z_F : Z_F \in \mathcal{Z}_F\big)$; $\bar{m}_C \leftarrow \operatorname{mean}\big(1 - Z_C : Z_C \in \mathcal{Z}_C\big)$ ▷ over formal and hybrid rollouts
    $\mu_F \leftarrow \operatorname{clip}\big(\mu_F + \eta(\bar{m}_F - \epsilon),\, 0,\, \mu_{\max}\big)$
    $\mu_C \leftarrow \operatorname{clip}\big(\mu_C + \eta(\bar{m}_C - \epsilon),\, 0,\, \mu_{\max}\big)$
  end for
end for
return π_θ

reinforcement learning does not drift too far from what supervision already taught. In the second stage we improve the policy with reinforcement learning. For every instance we draw G independent rollouts, run the verifier that each rollout selects, and turn the outcome into a scalar reward. From the G rewards of a group we compute an advantage for each rollout with the leave-one-out baseline (RLOO). The policy is then updated with a policy-gradient loss regularized by a KL penalty to the reference model. Finally, the two faithfulness constraints are enforced through their Lagrangian multipliers: after each update we measure how often the formal and hybrid routes were used on text that was not faithfully formalizable, and nudge the multipliers up or down so that this misuse stays within a small tolerance. The multipliers, Z_F, and Z_C used in this loop are defined in Algorithms 3 and 4.

C.2 Z_F SCORE CALCULATION ALGORITHM

Algorithm 3 computes Z_F, which estimates whether an instance can be faithfully written in the SMT-LIB form that the Z3 parser supports. We split the instance into two blocks, the premises passage and the hypothesis, and score each block on its own, because the formal route treats them differently and the hypothesis is the statement actually being checked. For a block we ask a language model to

Algorithm 2 StepReward(x, a): run the chosen route and score it

Require: instance x; chosen route a; gold label $y^\star$; duals μ_F, μ_C; cost weight λ; format bonus δ

$(\hat{y},\ \text{status},\ S) \leftarrow V(a, x)$ ▷ run route a; S is the SMT-LIB program if $a \in \{\text{FORMAL}, \text{HYBRID}\}$, else $\varnothing$

Faithfulness scores (formal or informal route):

if $a \in \{\text{FORMAL}, \text{HYBRID}\}$ **then**

  split x into blocks $\{b_1, b_2\} = \{\text{premises},\ \text{hypothesis}\}$

  $Z_F \leftarrow \min_i\ z_F(b_i)$ ▷ per-block formalizability; Algorithm 3

  if $S \neq \varnothing$ **then**

    $Z_C \leftarrow \big(\prod_i z_C(b_i, S)\big)^{1/2}$ ▷ geometric mean of back-translation similarity; Algorithm 4

  else

    $Z_C \leftarrow 1$ ▷ no program produced; nothing to check

  end if

else

  $Z_F \leftarrow 1; \quad Z_C \leftarrow 1$ ▷ informal route: no faithfulness term ($\alpha = 0$)

end if

Reward (Eq. in §E.3):

$\sigma \leftarrow +1$ if $a \neq \varnothing \wedge \hat{y} \neq \varnothing$, else -1 ▷ format bonus/penalty

$$\alpha \leftarrow \begin{cases} 1 & a = \text{FORMAL} \\ \alpha_H & a = \text{HYBRID} \\ 0 & a = \text{INFORMAL} \end{cases} \qquad \triangleright\ \alpha_H = 0.5$$

$r \leftarrow \delta\,\sigma + \mathbf{1}[\hat{y} = y^\star] - \lambda\, c_a - \alpha\big(\mu_F(1 - Z_F) + \mu_C(1 - Z_C)\big)$

return $r,\ Z_F,\ Z_C$

Algorithm 3 Z_F: formalizability of an instance

Require: instance x; annotator LLM; central weight w_c, peripheral weight w_p

split x into two blocks $\{b_1, b_2\} = \{\text{premises passage},\ \text{hypothesis}\}$

for each block $b \in \{b_1, b_2\}$ **do**

  Ask the LLM to list the distinct semantic *phenomena* in b; for each non-expressible phenomenon also return its centrality (CENTRAL / PERIPHERAL) using the prompt shown in figures 4, 7, and 8.

  $s_b \leftarrow 1$

  for each distinct phenomenon ϕ returned for b **do**

    if ϕ is not expressible in the supported SMT-LIB fragment **then**

      $s_b \leftarrow s_b - \big(w_c$ if ϕ is central else $w_p\big)$

    end if ▷ expressible phenomena cost nothing

  end for

  $s_b \leftarrow \max(0,\ s_b)$

end for

return $Z_F \leftarrow \min(s_{b_1}, s_{b_2})$ ▷ weakest block governs the whole

name the distinct semantic phenomena it contains using the prompt shown in figure 4, and, for any phenomenon that formal logic cannot represent, the same LLM predicts whether that phenomenon is central to the block's meaning or only peripheral. We start each block at a perfect score of one and subtract a penalty for every distinct phenomenon non-expressible in formal logic: a large penalty when it is central to the meaning and a small penalty when it is only incidental, while phenomena that the formal logic can express cost nothing. The score of the whole instance is the minimum of the two block scores, reflecting that a formalization is only as trustworthy as its least expressible part: if either the premises or the hypothesis cannot be captured in logic, the formal route should not be trusted even when the other block is clean.

C.3 Z_C SCORE CALCULATION ALGORITHM

Whereas Z_F asks whether an instance *could* be formalized, algorithm 4 computes Z_C, which checks whether the generated SMT-LIB representation for a given verification instance preserves the information/meaning. For this, we use the technique of *backtranslation*: For each block (the premise

Algorithm 4 Z_C: faithfulness of the generated formalization

Require: instance x; SMT-LIB program S produced by the formal route; back-translation LLM; sentence encoder $\mathrm{emb}(\cdot)$

split x into blocks $\{b_1, b_2\} = \{\text{premises},\ \text{hypothesis}\}$
split S into terms $\{t_1, t_2\} = \{\text{premise program},\ \text{goal}\}$ ▷ aligned 1:1 with the blocks
for $i = 1$ **to** 2 **do**
  if t_i is empty **then**
    $u_i \leftarrow 0$; **continue** ▷ content block but no formula: real failure
  end if
  $\hat{b}_i \leftarrow$ back-translate t_i into English via the LLM using the prompt shown in the figure 7.
  $u_i \leftarrow \frac{1}{2}\big(1 + \cos(\mathrm{emb}(b_i),\ \mathrm{emb}(\hat{b}_i))\big)$ ▷ cosine similarity, mapped to $[0, 1]$
end for
return $Z_C \leftarrow \big(\prod_i u_i\big)^{1/2}$ ▷ geometric mean over the two blocks

Table 3: Instances sampled per source dataset before route labelling.

Dataset	Sampled Instances
EntailmentBank	1,000
ProntoQA	1,500
ProofWriter	1,500
FOLIO	1,000
REVEAL	1,500
Total	6,500

passage and the hypothesis), we translate the logical part back into English with a language model and compare this back-translation to the original natural language block using the cosine similarity of their sentence embeddings, mapped into the unit interval. The overall Z_C is the geometric mean of the two per-block scores, so that a serious loss of meaning in either block pulls the whole score down rather than being averaged away.

D SUPERVISED FINE-TUNING (SFT)

The supervised stage teaches the router which verifier is appropriate for a reasoning step, using demonstrations built entirely offline. Each demonstration pairs a reasoning instance with (i) a *gold route* that is determined by construction rather than guessed, and (ii) a natural-language explanation of why that route fits the instance. This section describes the source datasets, how we turn them into verification instances, how the gold route is derived, the prompt used to generate explanations, and the statistics of the resulting dataset.

D.1 SOURCE DATASETS AND INSTANCE CONSTRUCTION

We draw reasoning steps from five datasets that together cover symbolic, deductive, and natural-language reasoning: EntailmentBank (`Task 1 Train` split) Dalvi et al. (2021), ProntoQA (`true-ontology 5hop` traces) Saparov & He (2023) in postorder, preorder, and random order, together with a (`fictional` variant split) provided by Pan et al. (2023b), FOLIO (`Train` split) Han et al. (2024), ProofWriter (`depth-5` split), and REVEAL Jacovi et al. (2024b).
From each dataset we build *step-level verification instances*. For an intermediate reasoning step, we form an instance whose hypothesis is that single step, together with whatever supporting information the dataset provides: a context passage, a sequence of prior steps, or both. For REVEAL we treat both kinds of steps as hypotheses: *logical* steps are handled as above, and for *attribution* steps we place the cited evidence in the context. We use the `follows`/ `not-follows` labels for each dataset's instance. After construction we sample a fixed number of instances per dataset to keep the mixture balanced across sources, as shown in Table 3.

D.2 DERIVING THE GOLD ROUTE

For every sampled instance we run both the formal route (translate to SMT-LIB and check with Z3) and the informal route (an LLM judge) exactly once, using `gpt-4.1-mini`. Let y be the ground-

Algorithm 5 Deriving the gold route for one instance

Require: instance x with gold step-validity y; verifier LLM
1: $\hat{y}_F \leftarrow$ label from the formal route on x ▷ may abstain ($\varnothing$)
2: $\hat{y}_I \leftarrow$ label from the informal route on x
3: **if** $\hat{y}_F \neq \varnothing$ **and** $\hat{y}_F = y$ **then**
4: **return** FORMAL
5: **else if** $\hat{y}_I \neq \varnothing$ **and** $\hat{y}_I = y$ **then**
6: **return** INFORMAL
7: **else**
8: **return** DROP ▷ no route matches y; excluded from SFT
9: **end if**

truth validity of the step. We assign the gold route by a formal-first rule: if the formal route returns a verdict equal to y, the gold route is *formal*; otherwise, if the informal route returns a verdict equal to y, the gold route is *informal*; if neither route reaches the correct verdict, the instance has no correct demonstration and is dropped from the supervised set. Algorithm 5 states this procedure.
The formal-first order reflects a deliberate preference: when a step can be settled by a sound logical proof, that proof is the more trustworthy signal, so the formal route is chosen whenever it is both applicable and correct. The informal route is used when formalization does not reach the right answer but natural-language judgement does. This construction is what makes the gold labels reliable: each demonstration records a route that *demonstrably* produced the correct verdict on that instance, rather than predicting the suitable route using an LLM.

D.3 Explanation Generation and Target Format

A route label alone does not teach the router *why* a route fits. For each kept instance we therefore generate a short, instance-specific explanation of the gold route using `Qwen2.5-14B-Instruct`. The model is shown the reasoning instance and told which route was selected, and is asked to justify that choice using only the characteristics of the instance—the kind of reasoning involved, whether the information can be faithfully formalized, and whether commonsense or world knowledge is required. Crucially, the explanation is generated *blind*: the model is explicitly forbidden from referring to any verifier's output, to SMT translations, to Z3 results, or to whether a route succeeded, so the explanation reflects only properties of the instance and never leaks the downstream verdict. The full instruction is given in the labelling prompt (Figure 9). The supervised target concatenates the explanation **E** and the route **R** into a single completion, `<think>` E `</think><route>` R() `</route>`, so that at test time the router produces its reasoning and its decision in the same format.

D.4 Train/Test Split and Balancing

We split the labelled instances into training and test sets with an $80/20$ ratio, stratified by (dataset, route) so that each source and each route is represented proportionally in both splits. We then balance *only the training set*: within each dataset we downsample the majority route so that the formal and informal counts match, which prevents the router from defaulting to whichever route happens to be more frequent in a given source. The test set is left in its natural, unbalanced distribution so that evaluation reflects the true prevalence of each route. The resulting SFT train and test dataset is shown in the Table 4. The final supervised dataset contains $2{,}808$ training instances and $1{,}138$ test instances.

D.5 Supervised Fine-Tuning Hyperparameters

The routing policy is a LoRA adapter on top of a frozen `Qwen2.5-7B-Instruct` base model. We train the adapter for 3 epochs with the AdamW optimizer at a learning rate of 1×10^{-4}, using a cosine learning-rate schedule with a warm-up ratio of 0.03. The per-device batch size is 2 with gradient accumulation of 8, giving an effective batch size of 16 sequences. Sequences are truncated on the left to a maximum length of 2048 tokens, so that the completion—and in particular the `</route>` tag the loss depends on—is never cut off. The LoRA configuration uses rank $r = 16$, scaling $\alpha = 32$, and dropout 0.05. The loss is a token-level cross-entropy computed over the completion only (the `<think>` reasoning together with the `<route>` decision); the prompt tokens are masked out. We append an end-of-sequence token to every completion so the model learns to

Table 4: Per-dataset route counts in the final SFT training and test splits. Training is balanced within each dataset; the test set is left unbalanced. **Fr.:** Formal, **Inf.:** Informal.

	Train		Test	
Dataset	Fr.	Inf.	Fr.	Inf.
EntailmentBank	68	68	18	168
FOLIO	341	341	86	89
ProntoQA	363	363	91	209
ProofWriter	680	194	171	49
REVEAL	195	195	49	208
Total	1,647	1,161	415	723

Table 5: Construction of the reinforcement learning set.

Dataset	**Target**	**Dropped**	**Needed**	**Available**
EntailmentBank	500	74	426	3,160
ProntoQA	500	2	498	13,500
ProofWriter	500	406	94	1,822
REVEAL	500	216	284	1,591
FOLIO	203	130	73	203
Total	**2,203**	**828**	**1,375**	**20,276**

stop after the route. All runs use seed 42, and gradient checkpointing is enabled to reduce activation memory.

E REINFORCEMENT LEARNING (RL)

E.1 REINFORCEMENT LEARNING DATASET

The reinforcement learning stage optimizes the router with a reward signal rather than gold routes, so it does not need routes labelled in advance and, importantly, must not reuse the instances the policy already saw during supervised fine-tuning. We therefore build the RL set entirely from instances that were *set aside* during SFT construction, keeping the two stages disjoint.

Two kinds of instances are available for reuse. The first are the *dropped* instances: steps for which neither the formal nor the informal route reached the correct verdict, so no gold route could be assigned (Section D.2). These were excluded from supervision precisely because they lack a demonstrably correct route, but that makes them useful for RL, where the policy must learn from a reward rather than imitate a fixed answer. The second are the *unused* instances: steps that were labelled during SFT construction but did not enter the final supervised set, either because they fell on the discarded side of the per-dataset balancing or because they were never sampled into it. Together these two pools form the candidate set for RL, and by construction none of them appears in the supervised training or test splits.

We assemble the RL set with a per-dataset target of 500 instances (FOLIO is capped at the 203 instances it has available). For each dataset we first include all of its dropped instances, then sample the remaining shortfall from that dataset's pool of unused instances, so that the harder, no-correct-route cases are retained in full and the rest of the quota is filled from the leftover labelled pool. Table 5 shows this breakdown: the target equals the number of dropped instances plus the number still needed, and the needed instances are drawn from a much larger available pool in every dataset except FOLIO, whose entire remaining pool is used. Duplicate instances are removed after merging, and we verify that the overlap with the supervised set is zero. The resulting RL set contains 2,203 instances across the five datasets out of which we took 1000 instances randomly to train the RL policy.

E.2 RL HYPERPARAMETER VALUES

The RL stage initializes both the trainable policy π_θ and the frozen reference π_{ref} from the same SFT adapter, so the KL penalty measures divergence from the fine-tuned model. Only the LoRA adapter

Table 6: Comparison of reasoning benchmarks. **Natural Context** indicates whether the benchmark uses naturally occurring contexts. **Human Ann. Label** indicates whether the labels are annotated by human experts. **Train Split** indicates whether the benchmark has train split. The **CoT Source** column indicates the origin of CoT annotations: PG = program-generated, LLM = LLM-generated, HW = human-written. † Human annotations are available only for a subset of the benchmark. ✓* in a column indicates that the corresponding property is satisfied through a combination of multiple sources or settings.

Dataset	Natural Context	Human Ann. Label	Train Split	CoT Source
ProofWriter	✗	✗	✓	PG
FOLIO	✓	✓	✓	✗
AR-LSAT	✓	✓	✓	✗
PrOntoQA	✗	✗	✓	PG
FaithCoT-Bench	✓	✓	✗	LLM
REVEAL	✓	✓	✗	LLM
EntailmentBank	✓	✓	✓	HW
SATBench	✗	✓†	✗	✗
BONAFIDE	✓*	✓†	✗	LLM
SIEVE (Ours)	✓*	✓	✓	✓*

is updated; the base model stays frozen. We use a group size of $G = 6$ rollouts per instance, sampled at temperature 1.0 with up to 768 new tokens per rollout, and a maximum sequence length of 2048. Optimization uses AdamW at a constant learning rate of 1×10^{-6} (no schedule, following common RL practice). The KL coefficient is $\beta = 0.05$ and the entropy coefficient is 0; the KL term uses the token-level k_3 estimator between the policy and reference adapters. We train for 1 epochs with a per-instance batch size of 2. To bound activation memory, each optimizer step processes the $B \times G$ rollouts in micro-batches of size 2 and accumulates gradients, which is exactly equivalent to a single full-batch update. The advantage estimator is RLOO (leave-one-out baseline). All runs use seed 42. The two faithfulness signals are enforced as inequality constraints via Lagrangian duals. The misuse tolerance is $\epsilon = 0.05$, the dual step size is $\eta = 0.1$, and both duals are clipped to $[0, \mu_{\max}]$ with $\mu_{\max} = 5.0$; their initial values are $\mu_F = \mu_C = 0.5$. After each optimizer step the duals are updated by projected gradient ascent on the average constraint violation (Algorithm 1, lines for μ_F, μ_C).

E.3 STEP REWARD FUNCTION

For every sampled trajectory, the policy receives the reward

$$R = \begin{cases} +\delta, & a \neq \varnothing \wedge \hat{y} \neq \varnothing, \\ -\delta, & \text{otherwise}, \end{cases} + \mathbf{1}[\hat{y} = y^*] \quad -\lambda c_a - \alpha(a)\Big[\mu_F(1 - Z_F) + \mu_C(1 - Z_C)\Big], \quad (7)$$

where $\hat{y}$ is the verifier prediction, y^* is the gold verification label, a is the selected route, and c_a is its computational cost. The first term is the formatting bonus: the policy receives $+\delta$ when it produces both a valid route a and a corresponding prediction $\hat{y}$, and $-\delta$ otherwise. The second term rewards a correct verdict. The third term charges for compute, scaled by λ. The final term is the latent-faithfulness penalty, applied through $\alpha(a)$, which equals 1 for the formal route, 0.5 for the hybrid route, and 0 for the informal route, so that the penalty acts only on the route that relies on symbolic reasoning. We set the formatting bonus to $\delta = 0.1$ and the cost weight to $\lambda = 0.1$. The per-route compute costs are $c_{\text{formal}} = 0.4$, $c_{\text{hybrid}} = 0.6$ and $c_{\text{informal}} = 0.2$. The strength of the faithfulness penalty is set by the learned duals μ_F and μ_C described in §E.2. The two faithfulness scores are aggregated over the two blocks of the instance. Z_F is the minimum over blocks—the weakest, least formalizable block governs the whole—computed from centrality-weighted phenomenon penalties with central weight 1.0 and peripheral weight 0.25 (Algorithm 3). Z_C is the geometric mean over blocks of the back-translation similarity (Algorithm 4).

F SIEVE DATASET

We construct a step-level verification dataset, SIEVE, from the CHECKWHY and FAITHCOT-BENCH datasets as follows: (1) For each claim in CHECKWHY, we prompt an LLM to generate

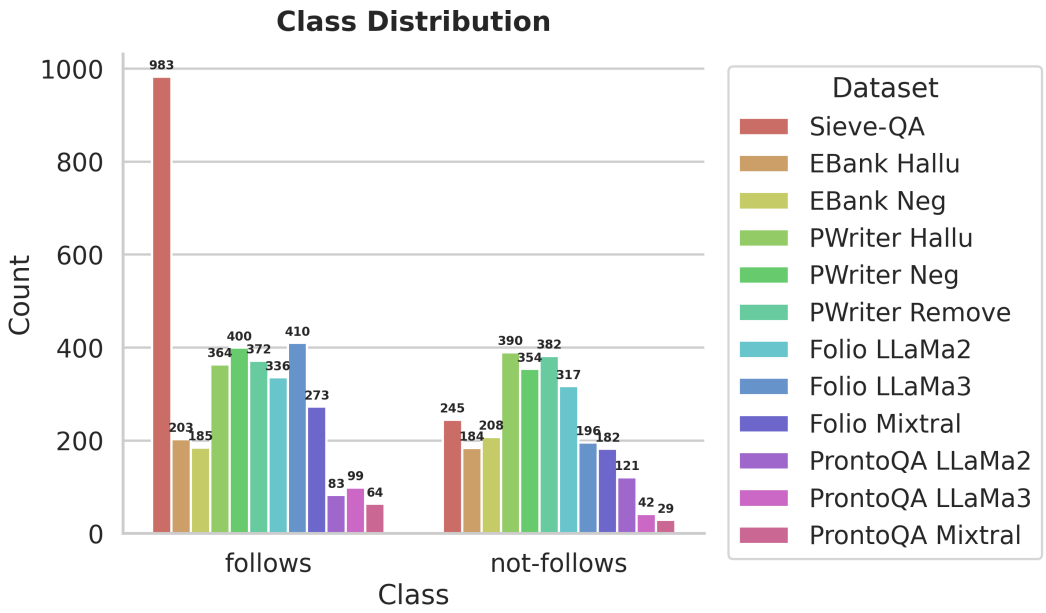

Figure 10: Per-subset distribution of the two labels. Most sources are close to balanced; SIEVE-QA is strongly skewed towards "follows" label, and the LLaMa3 / Mixtral variants of FOLIO and ProntoQA are moderately skewed.

chain-of-thought (CoT) reasoning and annotate each step as either an *attribution step* or a *logical step*. We then partition each reasoning chain at every logical step, producing multiple progressively extended instances, each terminating at a logical inference point. This results in the first variant, SIEVE-QA, containing 1,228 instances. (2) From FAITHCOT-BENCH, we use the chain-of-thought reasoning traces for the Logic-QA, TruthfulQA, and HLE benchmarks to construct the second variant, SIEVE-MCQ. For SIEVE-QA, the LLMs used to generate the CoT reasoning are **Phi-3.5-mini-instruct**, **Llama-3.1-8B-Instruct**, **Qwen2.5-7B-Instruct**, and **Llama-3.1-70B-Instruct**. Since SIEVE-QA is newly introduced in this work, we present its comparison with existing datasets; SIEVE-MCQ is used directly from FAITHCOT-BENCH without modification. We compare the twelve evaluation subsets along three axes: the balance of their labels, the length of their contexts, and the number of sentences per context. The subsets span four reasoning sources—SIEVE-QA, EntailmentBank (EBank), ProofWriter (PWriter), FOLIO (Folio), and ProntoQA—with several perturbation or generator variants each (e.g. hallucination, negation, removal, and the LLaMa2 / LLaMa3 / Mixtral generations). These instances differ sharply in shape, and those differences motivate several of our modelling choices.

F.1 LABEL BALANCE

Figure 10 shows the per-subset counts of the two labels, and the picture is consistent across most sources: EBank, PWriter, and FOLIO (LLaMa2) are close to balanced, with label entropies at or above 0.99 bits and imbalance ratios below 1.15. The exceptions are informative. SIEVE-QA is strongly skewed towards the positive label (983 vs. 245, an imbalance ratio of 4.0 and an entropy of only 0.72 bits), and the smaller ProntoQA and FOLIO variants generated by LLaMa3 and Mixtral are moderately skewed (imbalance ratios of 2.1–2.4). The direction of the skew is not uniform: Sieve-QA, FOLIO (LLaMa3), and ProntoQA (LLaMa3/Mixtral) lean positive, whereas ProntoQA (LLaMa2) leans negative.

F.2 CONTEXT LENGTH

The subsets differ most dramatically in how long their contexts are. Figure 11 shows the token-length distribution, and Figures 12 and 13 show the same data as a boxplot and a density curve. SIEVE-QA is the clear outlier: its mean context is 231 tokens, roughly 5.7× longer than the next-longest subset (FOLIO Mixtral, 40 tokens), and its length standard deviation of 734 tokens is more than an order of magnitude larger than any other subset. The boxplot (Figure 12) makes this con-

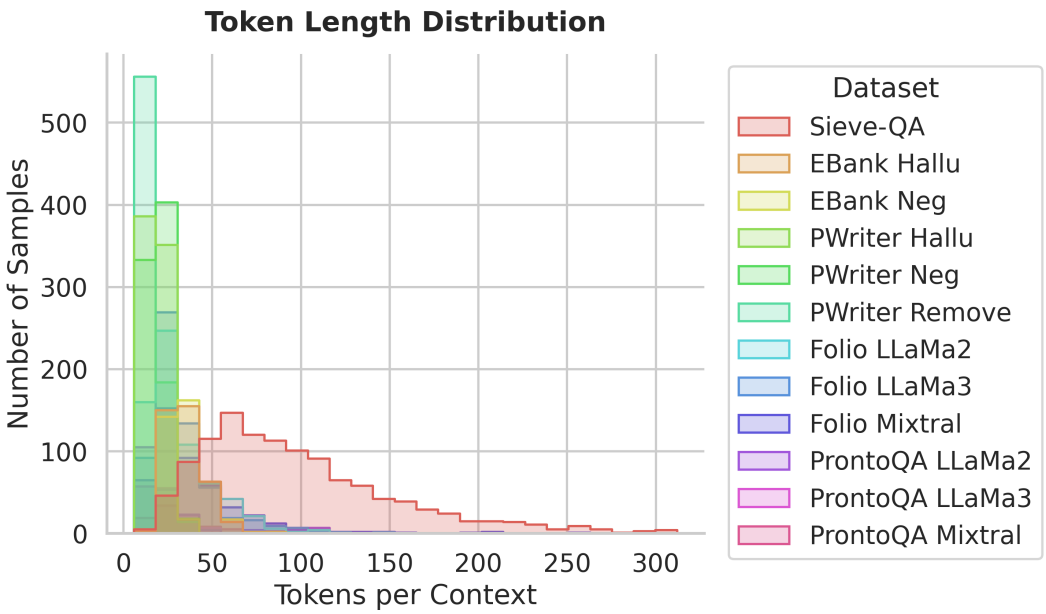

Figure 11: Token-length distribution per subset (clipped at the 99th percentile for readability). SIEVE-QA occupies a much wider and heavier range than every other subset, which cluster at short lengths.

crete: Sieve-QA has a long tail of outliers reaching past 6,000 tokens, while every other subset is compressed into a narrow band near the axis. The density plot (Figure 13), which shares a common horizontal scale, shows the same effect—every non-SIEVE subset collapses into a single spike near zero because their entire range is smaller than Sieve-QA's tail. At the other extreme, the ProofWriter variants are the shortest and most uniform: mean lengths of 15–19 tokens with standard deviations below 5. FOLIO and ProntoQA sit in between, at 24–40 tokens. This ordering matters for the formal route in particular: very short, templated contexts (ProofWriter) are the easiest to translate into logic faithfully, whereas SIEVE-QA's long, free-form passages are the least likely to be expressible in a compact SMT-LIB program. The long-context fraction is capped near 5% across subsets only because that column is defined relative to each subset's own 95th percentile; the absolute lengths behind it differ by more than an order of magnitude.

F.3 SENTENCE COUNT

The sentence-count views (Figures 14, 15, and 16) tell a similar but milder story. Most subsets contain 4–5 sentences per context on average, a narrow range that reflects their step-level construction: a context is typically a short premise set plus a single hypothesis. SIEVE-QA again has the heaviest tail—a mean of 11.8 sentences and outliers past 250 sentences per context (Figure 15)—while the remaining subsets rarely exceed a handful. The density view (Figure 16) shows the sharpest peak for EBank, consistent with its low sentence-count variance, and a slightly wider spread for SIEVE-QA and the Mixtral-generated subsets. Two points are worth drawing out. First, sentence count and token length do not rank the subsets identically: ProofWriter has a moderate sentence count but the shortest tokens, because its sentences are short and templated (about 4 tokens per sentence), whereas EBank has a similar sentence count but roughly twice the tokens per sentence. Second, the very high duplicate rates in the ProofWriter variants (up to 85% of contexts repeated, versus well under 1% for SIEVE-QA and ProntoQA) come from this templated construction: many ProofWriter contexts share the same short premise blocks.

F.4 TRAINING DATA VERSUS THE EVALUATION BENCHMARK

Having described the individual sources, we now compare the three pools the model actually sees as data: the supervised training set (SFT), the reinforcement learning set (RL), and the held-out SIEVE-QA benchmark on which we report results. These three pools are disjoint by construction

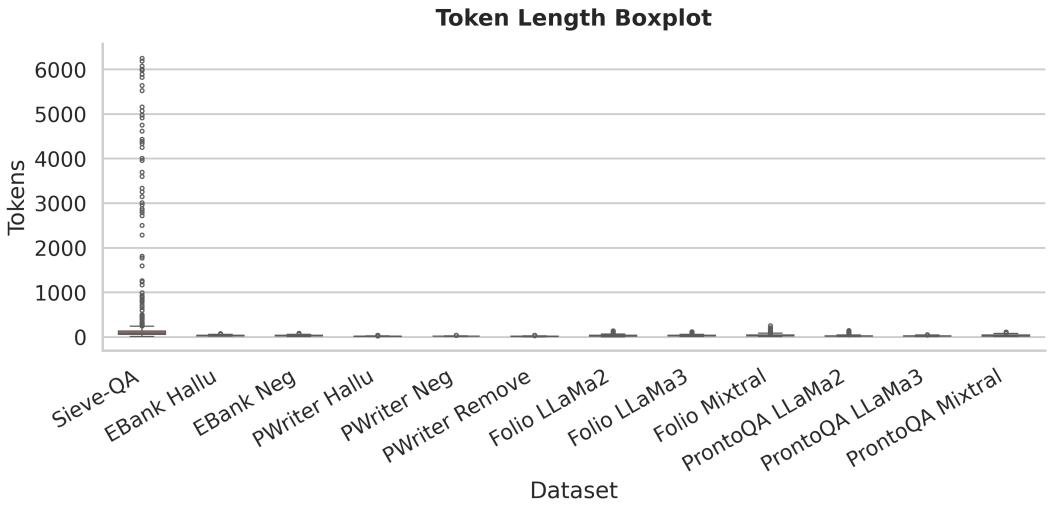

Figure 12: Token-length boxplot. SIEVE-QA shows an extreme high-length tail (outliers beyond 6,000 tokens); all other subsets are compressed near the axis, reflecting their short, templated contexts.

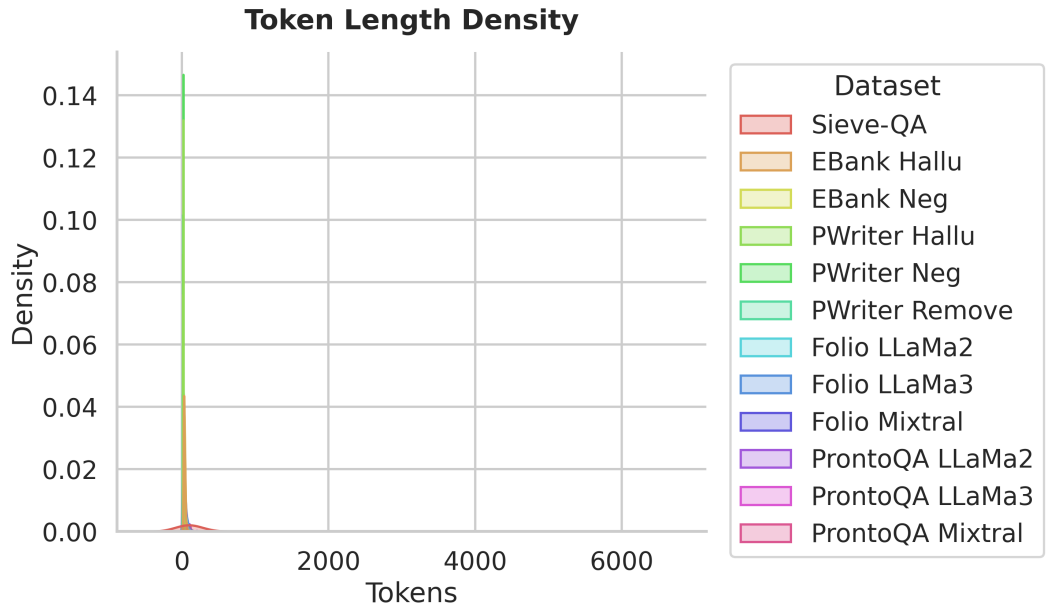

Figure 13: Token-length density on a shared scale. Because SIEVE-QA's range extends past 6,000 tokens, the other subsets appear as a single spike near zero.

(§D, §E.1), and comparing them shows what the model is trained on versus what it is tested on. We again look at label balance, context length, and sentence count.

F.4.1 LABEL BALANCE

Figure 17 shows the label distribution of the three pools. All three lean towards the *follows* label, but to very different degrees. The SFT set is the most balanced—1,746 *follows* against 1,062 *not-follows*, about 62% positive, with an entropy of 0.96 bits—which is expected, since we balanced the supervised training data by construction. The RL set is more skewed (709 vs. 291, roughly 71% positive), and the SIEVE-QA benchmark is the most skewed of the three (983 vs. 245, about 80% positive, entropy 0.72 and an imbalance ratio of 4.0). The progression matters for how results should

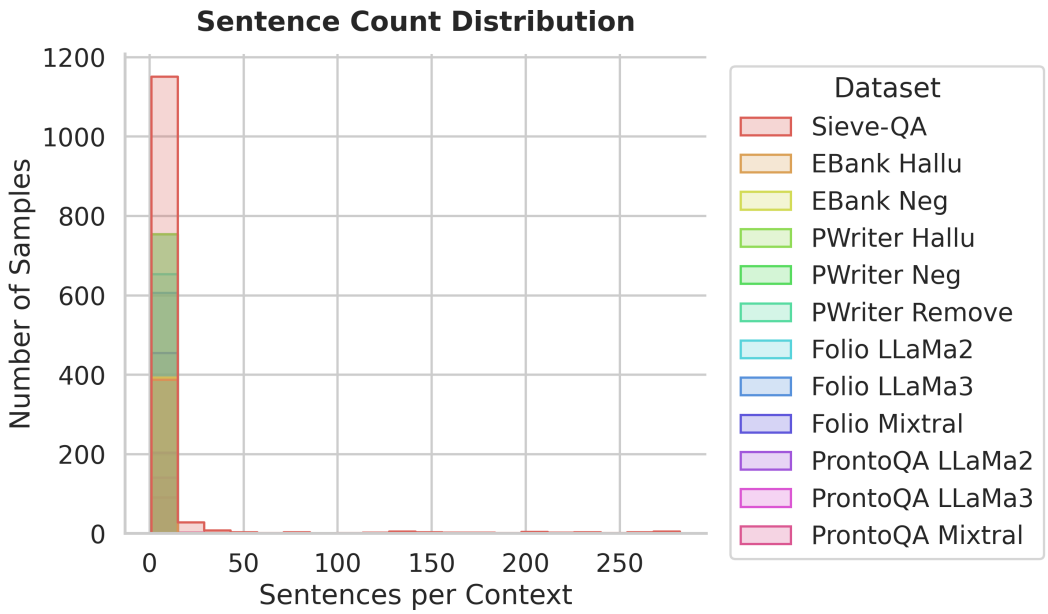

Figure 14: Sentence-count distribution per subset. Most contexts contain only a few sentences; SIEVE-QA carries a long tail of much larger contexts.

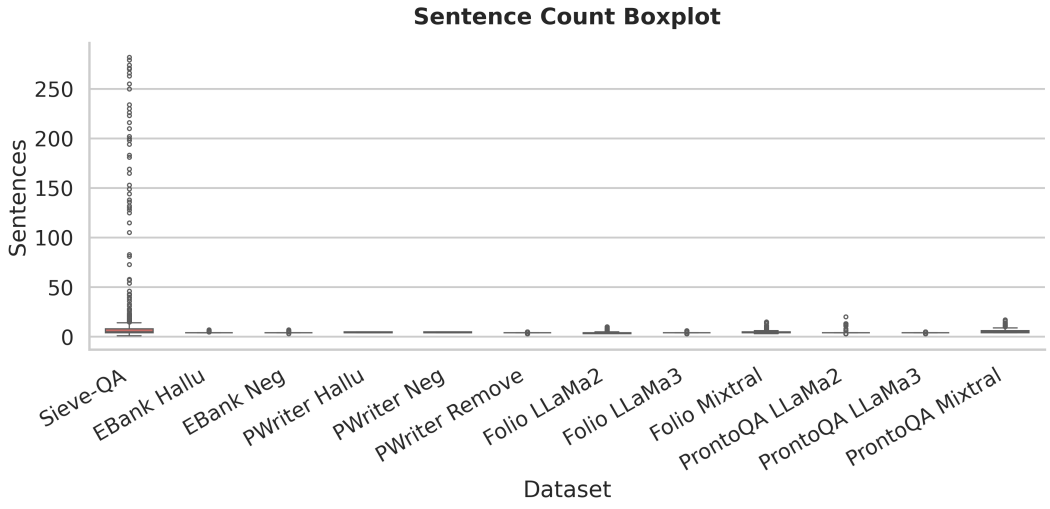

Figure 15: Sentence-count boxplot. SIEVE-QA reaches past 250 sentences per context; all other subsets stay within a few sentences, with only minor outliers.

be read: because the benchmark itself is positive-heavy, a model could reach roughly 80% accuracy by always predicting *follows*, so we report metrics that are not inflated by the majority class rather than accuracy alone.

F.4.2 CONTEXT LENGTH

The three pools differ sharply in context length, and this is the most important contrast in this section. Figure 18 shows the token-length histogram, with the boxplot and density views in Figures 19 and 20. The RL set is the shortest on average (42 tokens), the SFT set is intermediate (84 tokens), and the SIEVE-QA benchmark is by far the longest (231 tokens on average—about 2.8× the SFT set and 5.5× the RL set). More striking than the means is the spread: SIEVE-QA has a length standard deviation of 734 tokens and a tail reaching past 6,000 tokens (Figure 19), whereas both training

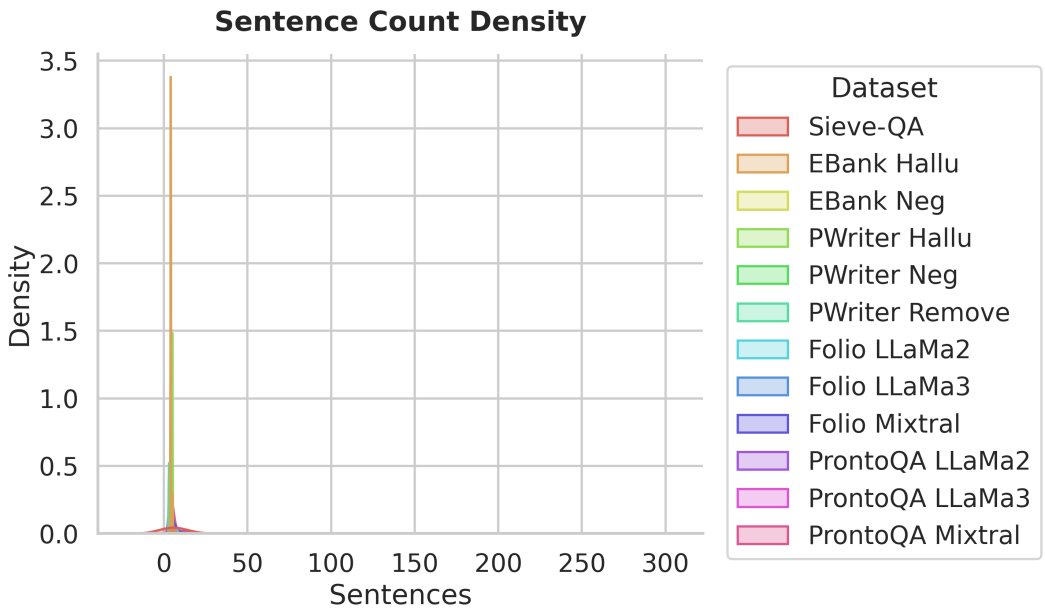

Figure 16: Sentence-count density on a shared scale. EBank shows the sharpest peak (lowest variance); SIEVE-QA and the Mixtral-generated subsets are slightly wider.

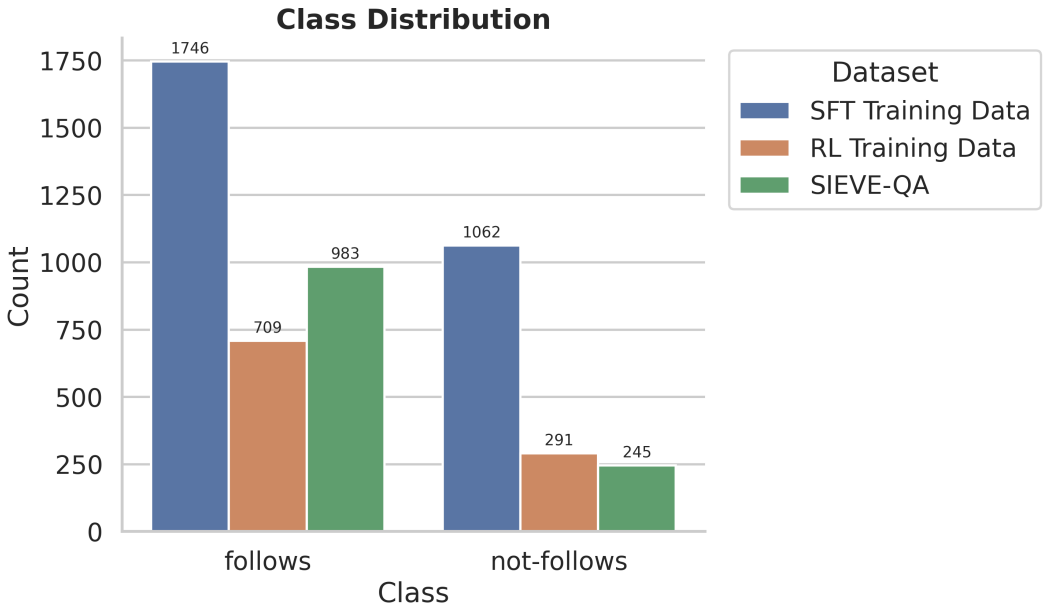

Figure 17: Label distribution of the three data pools. The SFT set is the closest to balanced; the RL set is moderately positive-skewed; and the SIEVE-QA benchmark is the most positive-skewed ($\approx 80\%$ *follows*).

pools stay within a few hundred. On the shared-scale density plot (Figure 20), the two training pools appear as sharp spikes near the origin while only SIEVE-QA has visible mass out in the long-context region.

F.4.3 SENTENCE COUNT

The sentence-count views (Figures 21, 22, and 23) refine the length picture. The SFT and SIEVE-QA pools have almost the same mean sentence count (11.9 and 11.8), yet their token lengths differ

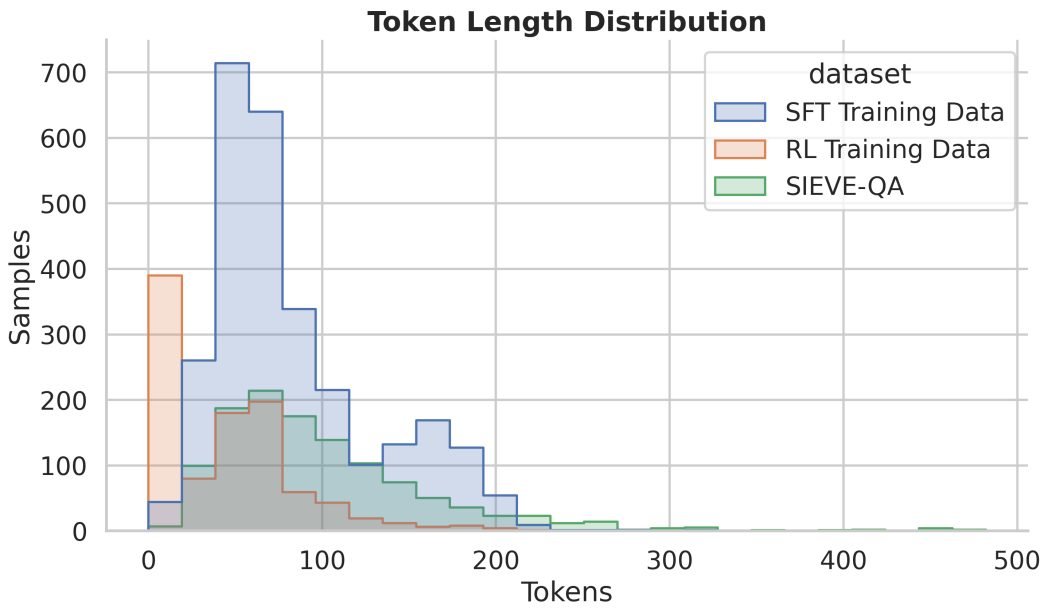

Figure 18: Token-length histogram for the three pools. The RL set concentrates at short lengths, the SFT set is intermediate, and SIEVE-QA carries a long right tail well beyond both training pools.

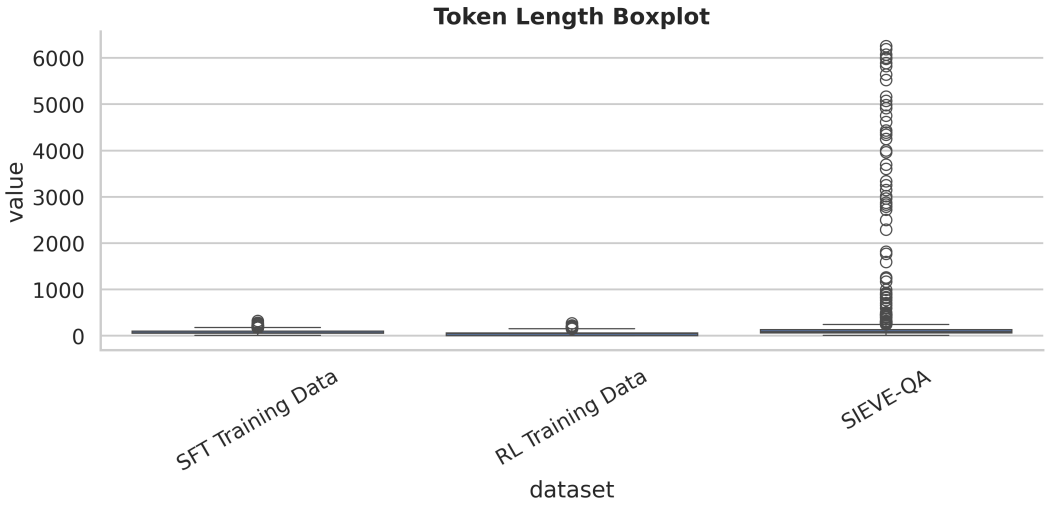

Figure 19: Token-length boxplot. SIEVE-QA shows an extreme high-length tail (outliers past 6,000 tokens); the SFT and RL pools are compressed near the axis by comparison.

by nearly 3×; SIEVE-QA simply has longer sentences and, crucially, a far heavier tail—its sentence-count standard deviation is 33 against 6.5 for the SFT set, and its boxplot (Figure 22) shows outliers past 250 sentences per context. The RL set has the fewest sentences on average (5.8). Its density curve (Figure 23) peaks sharply at the low end, which is consistent with a further property of the RL pool: about a third of its instances have an empty context field, because for those instances the reasoning to be checked lives entirely in the question and prior steps rather than in a separate context passage.

G GRAPH BASED APPROACH

Given a step s_i to be verified and the set of prior steps $\{s_1, \ldots, s_{i-1}\}$, we construct a directed graph $G = (V, E)$ where the node set $V = \{s_1, \ldots, s_{i-1}, s_i\}$ represents all steps up to and including the

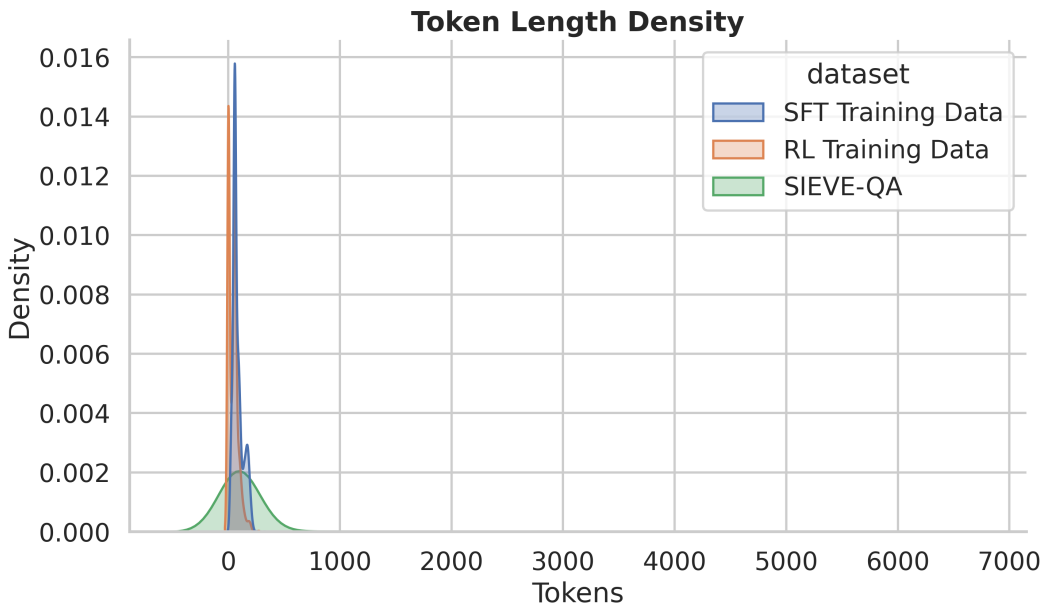

Figure 20: Token-length density on a shared scale. The two training pools form sharp peaks near zero, while only SIEVE-QA places noticeable mass at long lengths.

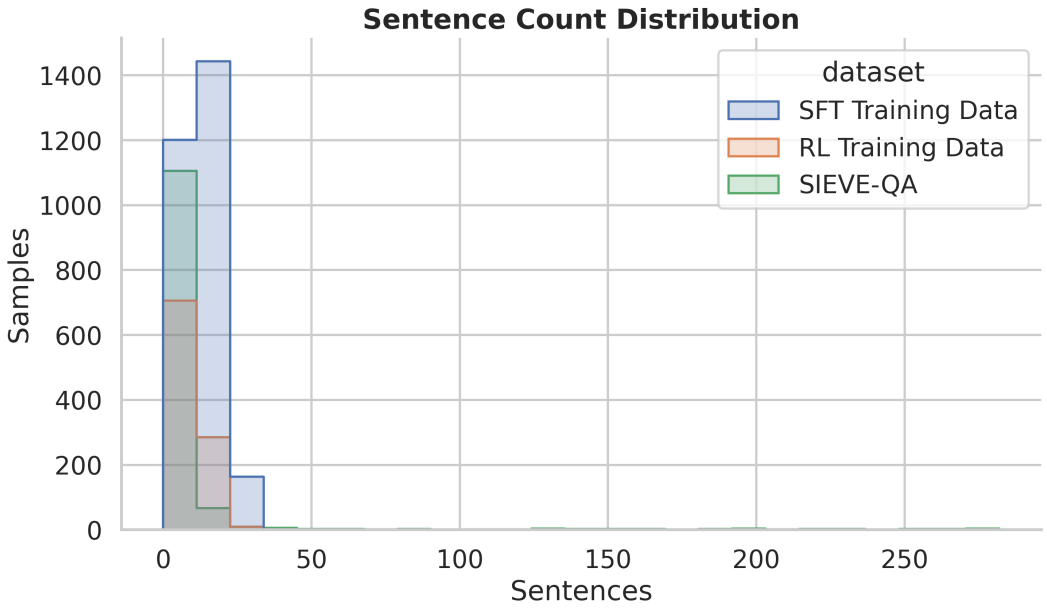

Figure 21: Sentence-count histogram. The RL pool concentrates at the fewest sentences; the SFT and SIEVE-QA pools are similar on average, but SIEVE-QA has a much longer tail.

step under verification. The graph has the following edges: A directed edge from any prior step s_j ($j < i$) to the step being verified s_i. These encode the possibility that s_j directly supports or is relevant to s_i. The labels associated with the edges are: the labels "*valid premise*"/ "*not valid premise*". The graph is serialized into a structured natural-language description and provided as input to the verifier LLM, along with the step to be verified.

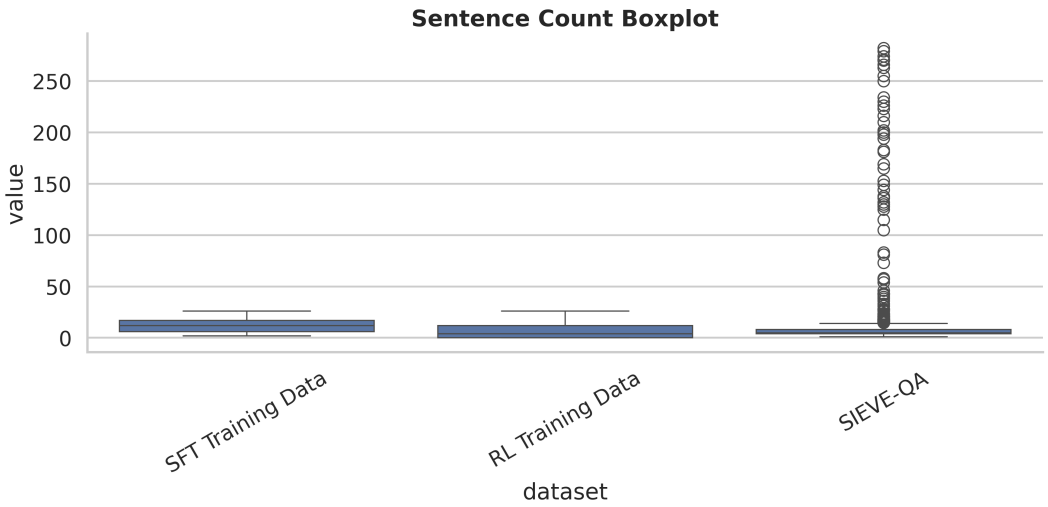

Figure 22: Sentence-count boxplot. SIEVE-QA reaches past 250 sentences per context; the SFT and RL pools remain within a much narrower range.

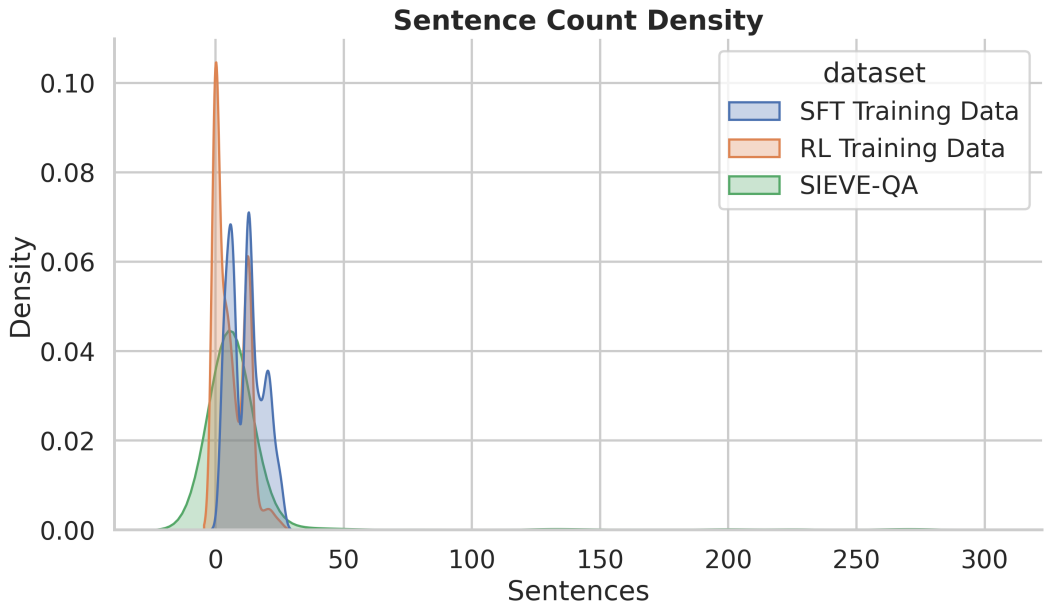

Figure 23: Sentence-count density on a shared scale. The RL pool peaks sharply at few sentences (in part because many of its contexts are empty), while the SFT and SIEVE-QA pools are slightly wider.