MORE THAN MEETS THE EYE: MEASURING LOGICAL REASONING BEYOND A SINGLE ACCURACY

Anonymous authors
Paper under double-blind review

ABSTRACT

An increasing number of large language models (LLMs) have achieved high accuracy on existing logical reasoning benchmarks, demonstrating impressive performance. However, producing the correct answer does not necessarily imply that a model can reliably identify and respond to the logical relations underlying a problem. Recent studies have highlighted the limitations of using overall accuracy as the primary measure of logical reasoning ability and have turned to examining models' reasoning behavior. In this work, we introduce **ReliLogic**, a behavioral reliability evaluation framework based on propositional-logic entailment questions. The framework consists of a Representation Set containing four logically equivalent forms and an Intervention Set constructed through logic-preserving and logic-altering interventions. Beyond final accuracy, ReliLogic evaluates cross-representation reliability, selective logical responsiveness, and instance-level prediction stability, providing a more comprehensive characterization of logical reasoning ability from the perspective of model behavior. We conduct a series of evaluations across 74 language-model configurations and uncover multiple failure modes that are not reflected by accuracy alone.

1 INTRODUCTION

Large language models (LLMs) have demonstrated remarkable capabilities and are increasingly deployed across diverse domains. Deductive and formal reasoning is particularly important because it decides whether LLMs can move beyond shallow statistical regularities to perform robust inference. Existing benchmarks typically assess the logical reasoning ability through task-level accuracy on mathematical problems, code-generation, and examination-style questions (Yu et al., 2020; Clark et al., 2020; Han et al., 2024). As frontier models are now approaching performance ceilings on many widely used benchmarks, their near-saturating scores are often taken as evidence of strong logical reasoning capabilities.

However, a fundamental question remains: *does high benchmark accuracy actually imply robust logical reasoning?* Growing evidence suggests that it does not. Prior work has shown that correct predictions may rely on lexical overlap, memorized reasoning or proof templates, and superficial token biases rather than the logical relations determining the answer (McCoy et al., 2019; Saparov & He, 2023; Jiang et al., 2024). Moreover, performance can degrade substantially under logically irrelevant changes, such as renaming symbols or adding irrelevant premises, even when the problem's underlying logic remain unchanged (Sanyal et al., 2022; Chen et al., 2024). These failures reveal a fundamental limitation of accuracy-based evaluation, that a model may achieve high accuracy while remaining sensitive to logically inconsequential variations. Reducing reasoning ability to a single accuracy score may therefore obscure important failure modes and conflate brittle surface-level pattern matching with robust formal deduction.

We argue that reliable logical reasoning requires more than high accuracy. At a minimum, a reliable inference model should exhibit invariance under representation changes, corresponding responsiveness to causal interventions, and stability across repeated evaluations. Existing studies have examined various aspects of LLM reasoning behavior, including robustness, sensitivity, and the validity of generated proofs (Stolfo et al., 2023; Tian et al., 2021). However, reliability across logically equivalent representations and instance-level stability across repeated runs have received comparatively limited attention.

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.

To address this gap, we introduce **ReliLogic**, a behavioral evaluation framework for reliable logical reasoning that assesses models along four dimensions, including accuracy, cross-representation reliability, selective causal responsiveness, and instance-level stability, as shown in Fig. 1. Specifically, ReliLogic is built on controlled propositional-logic entailment problems and consists of two components, the REPRESENTATION SET and the INTERVENTION SET. The REPRESENTATION SET expresses the same underlying logical relation in four logically equivalent forms, NATURAL, STRIPPED, CODE, COSMETIC-CODE, to test whether models can reliably recover the same logical relation despite changes in representation. The INTERVENTION SET applies two types of controlled interventions, those that alter the underlying logical relation and those that leave it unchanged. This design tests whether models respond appropriately when the logic changes while remaining invariant to interventions that are logically irrelevant. In addition, we repeatedly evaluate a subset of representative models on the same instances to assess whether they produce consistent predictions across runs.

Our evaluation covers 74 LLM configurations, including proprietary API-based models and open-weight models across a range of scales. We find that high aggregate accuracy does not necessarily imply reliable logical reasoning behavior. Specifically, some models with strong overall accuracy still produce a substantial number of inconsistent predictions across logically equivalent representations. Moreover, correctly solving an original problem does not guarantee that a model will respond appropriately when the underlying logical relation is altered, nor that it will preserve its prediction when the logic remains unchanged. We further observe that nearly identical aggregate accuracy across repeated runs can coexist with substantial prediction instability at the instance-level.

Our main contributions are as follows:

- We identify a key limitation of logical reasoning evaluations that rely predominantly on aggregate accuracy and articulate a set of behavioral properties required for more reliable logical reasoning. In particular, we jointly consider cross-representation reliability, causal responsiveness, and instance-level stability to provide a more complete characterization of model reasoning behavior.
- We introduce ReliLogic, a behavioral evaluation framework built on verifiable propositional-logic instances. By combining logically equivalent representations, controlled causal interventions, and repeated evaluations, ReliLogic evaluates logical reasoning ability in terms of directly testable behavioral properties.
- We systematically evaluate 74 LLM configurations and uncover multiple failure modes that are obscured by accuracy alone. High accuracy can coexist with failures across equivalent representations, inappropriate responses to logical interventions, and instance-level instability across repeated runs.

2 RELATED WORK

Logical reasoning benchmarks. Most benchmarks for evaluating the logical reasoning capabilities of LLMs primarily measure performance by accuracy. A number of studies (Yu et al., 2020; Liu et al., 2020; 2023; Wang et al., 2022) draw questions requiring deductive, conditional, and analytical reasoning from standardized examinations and evaluate models by final-answer accuracy. Another line of work constructs deductive reasoning tasks with well-defined ground-truth answers using formal rules. Clark et al. (2020) represent facts and rules in natural language and tests multi-step deduction by asking whether a query is entailed by a given theory. Han et al. (2024) further introduces human-authored natural-language problems paired with first-order logic formalizations, increasing logical complexity while providing more reliable ground-truth labels. Subsequent benchmarks (Parmar et al., 2024; Patel et al., 2024; Lin et al., 2025) further broaden the coverage of logical operators, inference patterns, rule combinations, and reasoning depths. These benchmarks have progressively expanded the scope and difficulty of logical reasoning tasks. Nevertheless, most still follow a similar evaluation paradigm, that a model's logical reasoning capability is characterized by its final accuracy.

Reasoning evaluations beyond accuracy. A growing body of work (Naik et al., 2018; McCoy et al., 2019; Gardner et al., 2020; Ribeiro et al., 2020) has shown that final-answer accuracy alone does not fully characterize reasoning behavior, as models may rely on shallow heuristics and remain brittle under targeted stress tests or semantic perturbations. Some studies, therefore, evaluate the reasoning

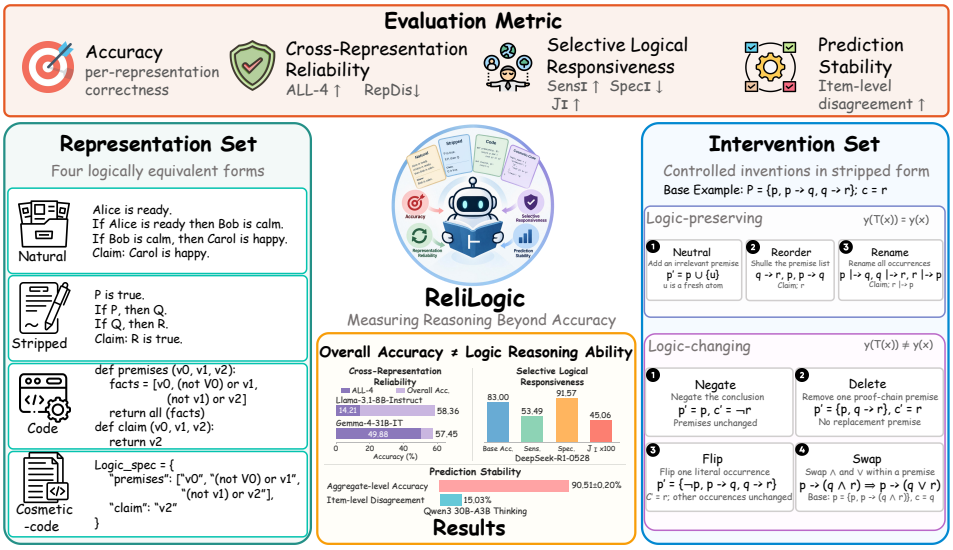

Figure 1: Overview of the ReliLogic evaluation framework. The figure illustrates the Representation Set with four logically equivalent input formats and the Intervention Set with logic-preserving and logic-changing transformations. It also summarizes the evaluation dimensions.

process itself. Tafjord et al. (2021) requires models to generate explicit proofs that can be verified. Tian et al. (2021) evaluates logical reasoning in terms of robustness, generalization, and proof traceability in addition to accuracy. Saparov & He (2023) maps generated chains of thought into symbolic proofs for formal analysis. Another line of work evaluates reasoning reliability through controlled interventions. Sanyal et al. (2022) uses logical contrastive edits and paraphrases to test sensitivity to logical changes and consistency under equivalent formulations. Model predictions are also affected by premise ordering (Chen et al., 2024) and token-level biases unrelated to the underlying logic (Jiang et al., 2024). Wu et al. (2024) modifies rules or structures to test whether models follow the intended reasoning relations rather than task-specific procedures. Stolfo et al. (2023) explicitly distinguishes robustness to irrelevant interventions from sensitivity to causally relevant ones. Together, these studies extend logical reasoning evaluation beyond final-answer correctness to proof processes, robustness, equivalence transformations, and causal interventions. However, most focus on specific failure modes or a limited subset of behavioral properties, leaving a systematic framework for evaluating logical reasoning across multiple complementary dimensions still lacking.

3 ReliLogic

3.1 Task Definition and Logical Instance Generation

Our evaluation framework is formulated as a binary entailment classification task over propositional logic. Let $\mathcal{A}$ denote a set of atomic propositions, and let $\mathcal{L}(\mathcal{A})$ denote the set of propositional formulas constructed over $\mathcal{A}$. A logical instance consists of a set of premises $P = \{p_1, \ldots, p_n\} \subset \mathcal{L}(\mathcal{A})$ and a target conclusion $c \in \mathcal{L}(\mathcal{A})$. The task is to determine whether c is logically entailed by P. Formally,

$$P \models c \iff \forall v,\ v \models P \Rightarrow v \models c, \tag{1}$$

where v denotes a truth assignment over the atomic propositions, and $v \models P$ indicates that every premise in P is satisfied under v. Accordingly, the label of an instance is defined as

$$y(P, c) = \begin{cases} \texttt{entailed}, & P \models c, \\ \texttt{not-entailed}, & P \not\models c. \end{cases} \tag{2}$$

We additionally require the premise set P to be satisfiable, that is, there must exist at least one truth assignment v such that $v \models P$. This constraint excludes instances with internally inconsistent premises. Under classical propositional logic, an inconsistent premise set vacuously entails any conclusion and therefore does not constitute a meaningful reasoning instance.

Logical instance generation. Data generation starts from base entailed instances for which a complete proof is available. We represent each base instance as

$$\mathcal{S} = (P, c, \pi, d, \rho, k), \tag{3}$$

where π denotes a valid proof path from the premises to the target conclusion, d is the proof depth, ρ records the sequence of inference rules used in the proof, and k denotes the number of distractor premises. The generator constructs base instances satisfying $P \models c$ using a predefined set of propositional inference rules, while explicitly controlling the proof depth and the number of distractor premises. The corresponding proof path and the instantiated rule sequence are recorded during generation. For each entailed instance, we construct a paired not-entailed negative instance. Negative instances are generated in a balanced manner using two strategies: flip, which modifies the target conclusion, and drop, which removes a critical premise required by the proof. During generation, a target entailment relation is specified for each instance: entailed instances must satisfy $P \models c$, whereas flip and drop instances must satisfy $P \not\models c$. Each candidate instance is then independently checked using SymPy (Meurer et al., 2017) and exhaustive truth-table evaluation. An instance is retained only if both verification procedures confirm the intended entailment relation, otherwise, it is regenerated.

Based on these verified instances, we construct two data components, REPRESENTATION SET and INTERVENTION SET, as shown in Fig. 1. REPRESENTATION SET derives multiple logically equivalent representations from each base logical instance. INTERVENTION SET, in contrast, independently constructs base instances together with semantics-preserving or semantics-altering variants. For each transformed instance, the actual entailment relation is again verified from the resulting premises and conclusion.

3.2 REPRESENTATION SET: FOUR LOGICALLY EQUIVALENT REPRESENTATIONS

The same logical relation can be expressed in natural language, abstract symbolic form, or code. If a model's prediction is reliably grounded in the underlying propositional logic, then its judgment should remain invariant when only the surface representation is changed while the logical relation is preserved. Motivated by this principle, we construct the REPRESENTATION SET, in which models are required to make the same entailment judgment across multiple representations of the same logical instance. This design allows us to examine whether model predictions remain consistent across representations.

For each logical instance $x = (P, c)$, we generate a corresponding version under each representation format s, denoted by $R_s(x)$. For every $s \in \mathcal{R}$, we require

$$\begin{gathered} y(R_s(x)) = y(x), \qquad \forall s \in \mathcal{R}, \\ y(x) \in \{\texttt{entailed}, \texttt{not-entailed}\}, \\ \mathcal{R} = \{\text{NATURAL}, \text{STRIPPED}, \text{CODE}, \text{COSMETIC-CODE}\}. \end{gathered} \tag{4}$$

All four representations are derived from the same logical instance. They differ only in the form of the input while preserving the same logical structure and generation conditions, and therefore do not alter the underlying entailment relation. The NATURAL representation expresses atomic propositions and logical relations in ordinary natural language. The STRIPPED representation performs symbolic abstraction by replacing semantically meaningful entities and properties with abstract symbols, while preserving the same logical operators and proof structure. The CODE representation expresses the Boolean relations through an executable solver-based framework. Finally, the COSMETIC-CODE representation encodes the exact same Boolean formulas in Python-like data structures, but without an executable solving framework. The REPRESENTATION SET contains 4,200 underlying logical instances, comprising 2,100 entailed, 1,050 flip, and 1,050 drop instances. Each instance is rendered in four logically equivalent representations, resulting in a total of 16,800 evaluation cases. This instance-level paired design allows us to evaluate not only whether a model produces the correct answer under each representation, but also whether its judgment remains consistent across logically equivalent representations.

3.3 INTERVENTION SET: CAUSAL INTERVENTIONS

Final-answer accuracy on a fixed dataset captures only a model's aggregate correctness on the given instances, but it does not reveal whether the model's predictions are actually determined by the

logical relation between the premises and the conclusion. To address this limitation, we construct the INTERVENTION SET, which applies controlled causal interventions to base logical instances and compares model predictions before and after intervention. We consider two types of interventions: logic-preserving and logic-changing interventions. logic-preserving interventions modify the input while leaving the logical label unchanged, which include adding neutral information, permuting the premises, and renaming symbols. logic-changing interventions alter the logical relation between the premises and the conclusion, thereby changing the corresponding label, which include negating the conclusion, removing a critical premise required by the proof, and reversing the logical polarity of a literal. Together, the two types of interventions test whether a model is sensitive to changes that genuinely affect the logical conclusion while remaining robust to changes that are logically irrelevant. Let the base instance be $x = (P, c)$, and let $T(x)$ denote the instance obtained by applying a causal intervention T to x. Let $y(x)$ denote the original label of x. We define the sets of logic-changing and logic-preserving interventions as

$$\mathcal{T}_{\mathrm{c}} = \{T \mid y(T(x)) \neq y(x)\}, \qquad \mathcal{T}_{\mathrm{p}} = \{T \mid y(T(x)) = y(x)\}. \tag{5}$$

The former tests whether a model changes its prediction appropriately when the underlying logical relation changes, whereas the latter tests whether the model preserves its prediction when the logical relation remains unchanged. A model with reliable logical reasoning should therefore exhibit sensitivity to interventions that alter the underlying logical relation and robustness to interventions that leave it unchanged. We refer to this joint requirement as *selective causal responsiveness*: the model should respond selectively to causally relevant changes while remaining invariant to causally irrelevant ones. The INTERVENTION SET uses the STRIPPED representation to minimize semantic cues. After each intervention, we recompute the entailment relation from the transformed premises P' and conclusion c', and apply the same verification procedure described in Section 3.1 to determine the ground-truth label. Transformed instances whose verified labels do not match the intended intervention outcome are discarded. The INTERVENTION SET contains 2,365 cases in total, comprising 400 base instances, 765 logic-changing cases, and 1,200 logic-preserving cases.

3.4 BEHAVIORAL MEASUREMENTS

The REPRESENTATION SET and INTERVENTION SET evaluate model behavior under logically equivalent representations and causal interventions, respectively. Building on these two evaluation settings, we define metrics to quantify cross-representation logical reliability and selective causal responsiveness to logical interventions.

For the REPRESENTATION SET, in addition to reporting accuracy under each of the four representations, we focus on whether the same logical instance is answered correctly across all representations and whether the model's prediction changes under representational variation. For a model m, let $z_{mis} \in \{0, 1\}$ indicate whether the model correctly classifies logical instance i under representation s. We define the cross-representation joint accuracy, denoted as ALL-4, as

$$\text{ALL-4}(m) = \frac{1}{N} \sum_{i=1}^{N} \prod_{s \in \mathcal{R}} z_{mis}. \tag{6}$$

This metric measures the proportion of underlying logical instances that are answered correctly under all four representations. We additionally report Representation Disagreement(RepDis):

$$\text{RepDis}(m) = \frac{1}{N} \sum_{i=1}^{N} \mathbf{1}\left[|\{\hat{y}_{mis} : s \in \mathcal{R}\}| > 1\right], \tag{7}$$

where $\hat{y}_{mis}$ denotes the predicted label of model m for instance i under representation s. This metric measures the proportion of logical instances for which the model produces different predictions across the four representations, thereby characterizing the sensitivity of its predictions to representational changes. While ALL-4 measures joint correctness across representations, RepDis measures invariance of the predictions themselves.

For the INTERVENTION SET, we separately evaluate whether models respond correctly to logic-changing and logic-preserving interventions. We define Intervention Sensitivity($\mathrm{Sens_I}$) and Intervention Specificity($\mathrm{Spec_I}$). Unlike conventional notions of sensitivity and specificity, both metrics are conditioned on the model having correctly classified the corresponding base instance:

$$\mathrm{Sens_I} = \Pr(\hat{y}(T(x)) = y(T(x)) \mid \hat{y}(x) = y(x),\ T \in \mathcal{T}_{\mathrm{c}}), \tag{8}$$

Table 1: The experiment results on the REPRESENTATION SET and the INTERVENTION SET. All values except J_I are percentages.

Model	REPRESENTATION SET							INTERVENTION SET		
	Natural↑	Stripped↑	Code↑	Cosmetic↑	Overall↑	ALL-4↑	RepDis↓	Sens_I↑	Spec_I↑	J_I↑
GPT-5	99.90	**99.98**	**100.00**	**99.98**	99.97	99.86	0.14	**100.00**	**100.00**	**1.00**
Qwen3.5-397B-A17B	**99.98**	**99.98**	99.98	**99.98**	**99.98**	**99.90**	**0.10**	**100.00**	**100.00**	**1.00**
Gemini 3.5 Flash	95.98	96.38	99.88	99.86	98.03	92.74	7.26	94.97	97.86	0.93
Qwen3-30B-A3B Thinking	80.05	85.98	98.19	99.83	91.01	68.69	31.31	82.94	86.86	0.70
DeepSeek-R1-0528	91.19	87.74	77.71	75.40	83.01	59.90	36.95	53.49	91.57	0.45
Llama-3.1-8B-Instruct	69.24	62.76	50.86	50.57	58.36	14.21	82.98	56.34	67.60	0.24
Gemma-4-31B-IT	64.55	64.79	50.26	50.19	57.45	49.88	16.50	8.61	97.35	0.06
Qwen2.5-Coder-14B	77.60	73.45	54.74	55.10	65.22	34.55	58.81	60.21	75.33	0.36
Gemma-4-26B-A4B-IT	80.17	80.36	50.10	50.12	65.19	34.98	59.02	73.54	86.12	0.60

$$\text{Spec}_\text{I} = \Pr(\hat{y}(T(x)) = y(T(x)) \mid \hat{y}(x) = y(x),\ T \in \mathcal{T}_\text{p})\,. \tag{9}$$

Sens_I measures the proportion of correctly solved base instances for which the model also adjusts its prediction correctly after an intervention changes the underlying logical relation. In contrast, Spec_I measures the proportion for which the model preserves a correct prediction when the intervention leaves the underlying logical relation unchanged. Following the form of Youden's J statistic (Youden, 1950), we combine the two measures as

$$J_\text{I} = \text{Sens}_\text{I} + \text{Spec}_\text{I} - 1. \tag{10}$$

Here, $J_\text{I} \in [-1, 1]$, with higher values indicating a stronger ability to selectively respond to logical interventions: the model correctly changes its judgment when the underlying logical relation changes while preserving its correct judgment when the logical relation remains unchanged. Moreover, we define Item-level Disagreement to evaluate prediction stability across repeated runs. Let $\hat{y}_{mir}$ denote the predicted label of model m for instance i in evaluation run r, and let $\mathcal{R}_\text{run}$ denote the set of repeated runs,

$$\text{Item} - \text{level Disagreement} = \frac{1}{N}\sum_{i=1}^{N} \mathbf{1}\left[|\{\hat{y}_{mir} : r \in \mathcal{R}_\text{run}\}| > 1\right]. \tag{11}$$

4 EXPERIMENTS AND ANALYSIS

4.1 EXPERIMENTAL SETUP

We evaluate 74 model configurations spanning proprietary API-based and open-weight models. Local models are evaluated with released BF16 or FP8 weights on H100 GPUs, while hosted models are accessed through their APIs. Full model configuration list are provided in the Appendix. B. A frozen deterministic parser maps each response to `entailed`, `not-entailed`, or `UNPARSED`, with unparseable responses counted as incorrect.

4.2 CORRECTNESS IS NOT CROSS-REPRESENTATION RELIABILITY

Table 1 reports results on the REPRESENTATION SET and INTERVENTION SET for representative model configurations (Singh et al., 2025; Qwen Team, 2026; Google DeepMind, 2026b; Yang et al., 2025; Hui et al., 2024; Gemma Team et al., 2026; Grattafiori et al., 2024; Guo et al., 2025). Although several state-of-the-art models approach the performance ceiling, our evaluation still reveals substantial weaknesses remained for many models.

To illustrate cross-representation inconsistency, Fig. 6 (a) presents a representative example of Qwen3-30B-A3B Thinking. Although the four input forms encode exactly the same logical relation, the model produces a different prediction for STRIPPED. Qwen3-30B-A3B-Thinking achieves 91.01% overall accuracy but only 68.69% on ALL-4, with 31.31% RepDis, showing that high accuracy can coexist with fragile judgments under equivalent representations. The contrast between Llama-3.1-8B-Instruct and Gemma-4-31B-IT further demonstrates this point. Despite similar overall accuracy, Llama-3.1-8B-Instruct achieves only 14.21% ALL-4 with 82.98% RepDis, whereas

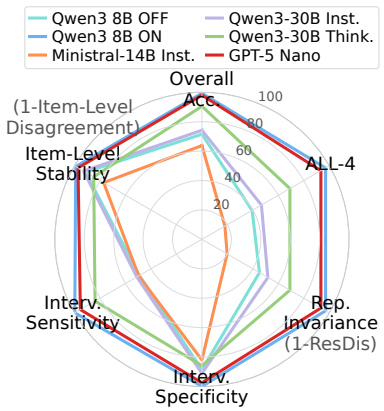

Figure 2: Reasoning ability profiles of representative model configurations.

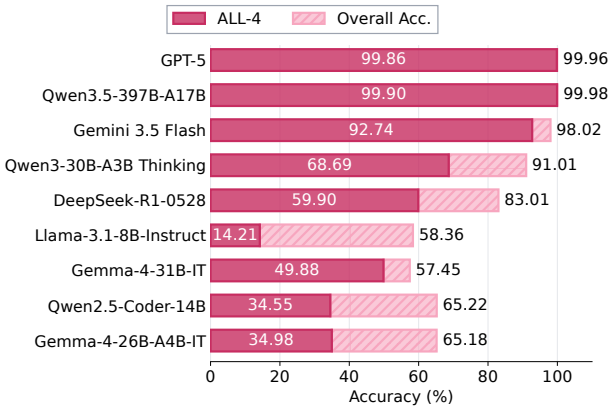

Figure 3: Overall accuracy and ALL-4 comparison on THE REPRESENTATION SET.

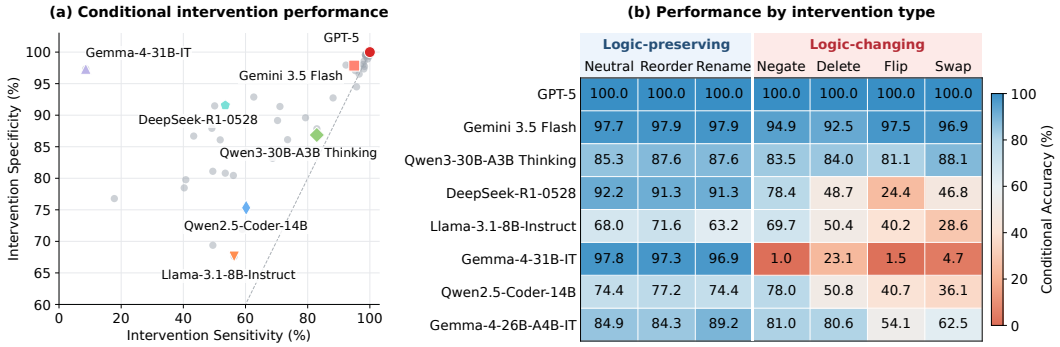

Figure 4: Detailed performance on THE REPRESENTATION SET. Panel (a) plots Intervention Sensitivity against Intervention Specificity for all 74 models. Panel (b) reports conditional accuracy for all types of logic-perserving and logic-changing interventions.

Gemma-4-31B-IT reaches 49.88% on ALL-4 with 16.50% RepDis.Similar overall accuracy can therefore correspond to fundamentally different degrees of representation invariance.

These failures are concentrated mainly in the CODE representation. Among the 74 configurations, 49 perform worse on CODE than on NATURAL, with an average drop of 5.87%, and CODE is the weakest representation for 33 configurations. In contrast, the aggregate difference between NATURAL and STRIPPED is near zero. Removing the executable wrapper provides little recovery. COSMETIC-CODE improves over CODE by only 0.65% on average. These results suggest that models often preserve logical performance after lexical semantics are removed but become more brittle when the same structure is expressed as code-like Boolean expressions.

Overall, correct answers do not imply reliable recognition of the same logical relation across equivalent representations. Cross-representation reliability requires stable judgments when the underlying logic is unchanged, making overall accuracy alone insufficient to characterize this capability.

4.3 SELECTIVE LOGICAL RESPONSE FAILURE UNDER CAUSAL INTERVENTION

As shown in Table 1, accuracy alone is insufficient to reflect whether a model can correctly and selectively update its prediction when an intervention changes the underlying logical relation. A representative example is DeepSeek-R1-0528, which achieves an accuracy of 83.00% on the base instances but only 53.49% $\mathrm{Sens}_{\mathrm{I}}$, failing to correctly update nearly half of its initially correct predictions after logic-changing interventions. Moreover, models with similar accuracy also differ substantially in responsiveness. Qwen2.5-Coder-14B and Gemma-4-26B-A4B-IT achieve nearly identical overall accuracy and ALL-4, yet the latter has a J_{I} score 0.24 higher, indicating stronger

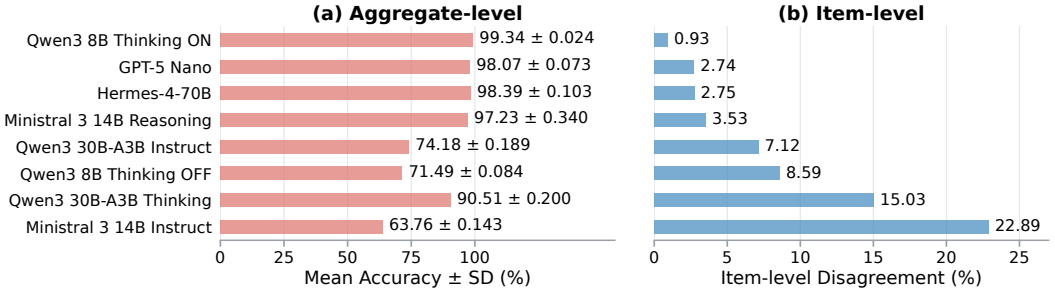

Figure 5: Prediction reproducibility across repeated evaluation runs. Panel (a) reports the mean aggregate accuracy and standard deviation, while panel (b) reports the corresponding item-level prediction disagreement rates.

tracking of logical changes. Fig. 6 (b) presents an example of two types of intervention-response failure. The model correctly answers the base instance, but fails to update its prediction after a delete or reorder operation.

Intervention Sensitivity and Intervention Specificity further reveal three distinct failure modes. The first consists of models with both low $\mathrm{Sens}_{\mathrm{I}}$ and low $\mathrm{Spec}_{\mathrm{I}}$, which exhibit broad instability in their judgments. For example, the base accuracy of Llama-3.1-8B-Instruct is 62.50%, while its $\mathrm{Sens}_{\mathrm{I}}$ and $\mathrm{Spec}_{\mathrm{I}}$ are only 56.34% and 67.60%, respectively, yielding behavior close to random guessing. The second failure mode consists of models with high $\mathrm{Spec}_{\mathrm{I}}$ but low $\mathrm{Sens}_{\mathrm{I}}$, which may arise when a model tends to preserve the same answer regardless of how the input changes. Gemma-4-31B-IT exhibits this pattern, with a $\mathrm{Spec}_{\mathrm{I}}$ of 97.35% but a $\mathrm{Sens}_{\mathrm{I}}$ of only 8.61%. It predicts `entailed` for more than 90% of the questions. Third, high $\mathrm{Sens}_{\mathrm{I}}$ but low $\mathrm{Spec}_{\mathrm{I}}$ corresponds to overreacting to input changes regardless of their logical relevance, although none of the 74 configurations exhibits a pronounced instance of this pattern.

4.4 Stable Scores, Unstable Predictions

Stable aggregate scores do not imply that a model makes consistent judgments on the same instances across repeated runs. In fact, aggregate-level reproducibility and item-level reproducibility capture distinct properties. To examine this distinction, we conduct three complete evaluation runs for eight representative model configurations. As shown in Fig. 5, all configurations exhibit highly stable aggregate accuracy. The standard deviation across the three runs is at most 0.34% and as low as 0.01%. In contrast, item-level disagreement ranges from 0.93% to 22.89% across configurations. This indicates that even when aggregate scores remain nearly unchanged across repeated runs, the specific instances on which the model makes errors can vary substantially. Fig. 6 (c) illustrates an example in our experiment. Given exactly the same input, the model produces different judgments across repeated evaluation runs. Therefore, consistency of item-level predictions provides a more direct measure of the stability of a model's reasoning judgments than aggregate-score variance alone.

4.5 Errors Associate With Instance Structure

Further analysis reveals that model errors are not uniformly distributed but are closely associated with the logical structure of the instances. In particular, models are more prone to errors on problems where critical evidence is missing than on those where the logical relation is explicitly altered. In addition, increasing proof depth leads to substantially larger performance degradation for models that have not yet reached saturation.

Models are particularly vulnerable to missing critical evidence. As shown in Fig. 7, among all `not-entailed` instances in the Representation Set, 51 of the 74 evaluated configurations perform worse on Drop than on Flip, with an average gap of 2.93%. This gap widens to 5.25% under the Natural representation, while remaining at approximately 2% under Stripped, Code, and Cosmetic-Code. These results suggest that many models more readily reject a conclusion whose logic is explicitly disrupted than those made unsupported by removing critical evidence, especially in natural language. Increasing proof depth also makes the task more challenging, but its effect varies substantially across models. The most pronounced degradation is concentrated among

Qwen3-30B-A3B Thinking

(a) Four Representations

Label: not-entailed

Natural Model: not-entailed

```
1. Fred is quiet.
2. If Hank is not happy,
then the dog is not tired.
Claim: Gina is not empty.
```

Code Model: not-entailed

```
def premises(v0, v1, v2, v3):
    facts = [v0,
        ((not (not v2))
        or (not v3))]
    return all(facts)
def claim(v0, v1, v2, v3):
    return (not v1)
```

Stripped Model: entailed

```
1. P is true.
2. If not R, then not S.
Claim: Q is false.
```

Cosmetic Model: not-entailed

```
logic_spec = {"premises": [
 "v0", "((not (not v2)) "
 "or (not v3))"],
 "claim": "(not v1)"}
```

(b) Causal Interventions

Delete

Original

```
1. If U, then T.
2. P is true.
3. If not Q, then not R and S.
4. If P, then not Q.
5. If T, then R.
Claim: U is false.
```

Label: entailed
Model: entailed

After delete

```
1. If T, then R.
2. If P, then not Q.
3. If not Q, then not R and S.
4. P is true.
Claim: U is false.
```

Label: not-entailed
Model: entailed

Reorder

Original

```
1. P is true.
2. If not T, then U.
3. If P, then not Q.
4. If not Q, then not R.
Claim: S is false.
```

Label: not-entailed
Model: not-entailed

After reorder

```
1. If not Q, then not R.
2. If not T, then U.
3. P is true.
4. If P, then not Q.
Claim: S is false.
```

Label: not-entailed
Model: entailed

(c) Repeated Runs

```
1. The clerk is green.
2. Either the driver is wet, or the artist is not sharp.
Claim: the nurse is not heavy.
```

Run 1	Run 2	Run 3
Model: entailed	Model: not-entailed	Model: not-entailed

Figure 6: An example of Qwen-30B-A3B Thinking. (a) Prediction inconsistency across logically equivalent representations; (b) inappropriate response to a controlled logical intervention; and (c) prediction disagreement across repeated evaluations. The three panels use different instances and illustrate the behavioral quantities measured by ReliLogic.

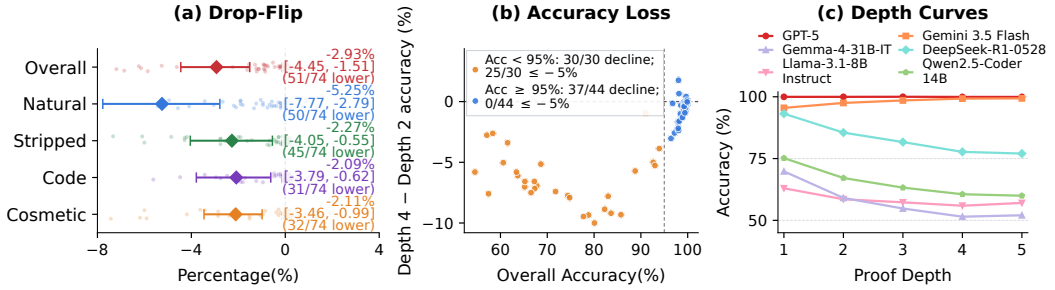

Figure 7: Performance of negative-instance construction and proof depth. Panel (a) shows the accuracy difference between Drop and Flip instances across the overall dataset and four representations. Panel (b) plots the change in accuracy from proof depth 2 to depth 4 against overall accuracy across model configurations. Panel (c) shows accuracy at proof depths 1–5 for representative models.

models that have not yet reached performance saturation. When proof depth increases from 2 to 4, accuracy decreases for 67 of the 74 evaluated configurations. More strikingly, all 30 configurations with overall accuracy below 95% exhibit a decline, and 24 of them drop by at least 5%. By contrast, among the 44 configurations with overall accuracy of at least 95%, 37 also show some degradation, but none decline by more than 5%. Taken together, these results indicate that reasoning errors are systematically associated with the structural properties of logical instances rather than being uniformly distributed across examples. **More Experiments and Analysis.** More experiment results and analysis are reported in Appendix. B.

5 Conclusion

We introduce ReliLogic, a logic reasoning evaluation framework that assesses LLMs beyond accuracy along three dimensions, cross-representation reliability, selective responses to logical interventions, and instance-level stability. Our results show that accuracy alone does not fully capture the logical behavior exhibited by a model or its actual logical reasoning capability. Accuracy remains an important metric for evaluating logical reasoning, but it is insufficient on its own to determine whether a model can reliably make predictions in accordance with the underlying logical relations. These findings suggest that future evaluations should look beyond final scores and place greater emphasis on the behavioral properties reflected in model predictions. By exposing reasoning failures that remain hidden under aggregate accuracy, our work provides a foundation for more comprehensive evaluations of logical reasoning in future.

AI USE STATEMENT

Generative AI systems are used only as limited aids for editorial and implementation tasks, including code drafting and debugging, language refinement, and manuscript organization. The authors are responsible for designing the framework and experiments, evaluating the scientific interpretations, and making the final decisions regarding all claims. The LLMs mentioned serve as experimental systems in this study. We review and test AI-assisted code to ensure that it performs the intended computations, verify numerical statements against the recorded outputs, and carefully review and revise all AI-assisted prose. The authors therefore retain full responsibility for the final text, mathematical content, results, software, and accompanying materials.

REFERENCES

Xinyun Chen, Ryan Andrew Chi, Xuezhi Wang, and Denny Zhou. Premise order matters in reasoning with large language models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 6596–6620. PMLR, 21–27 Jul 2024.

Peter Clark, Oyvind Tafjord, and Kyle Richardson. Transformers as soft reasoners over language. In Christian Bessiere (ed.), *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pp. 3882–3890. International Joint Conferences on Artificial Intelligence Organization, 7 2020. doi: 10.24963/ijcai.2020/537. Main track.

Matt Gardner et al. Evaluating models' local decision boundaries via contrast sets. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pp. 1307–1323, Online, November 2020. Association for Computational Linguistics.

Gemma Team et al. Gemma 4 technical report, 2026. URL https://arxiv.org/abs/2607.02770.

Google DeepMind. Gemini 3.1 pro: Model card. https://deepmind.google/models/model-cards/gemini-3-1-pro/, February 2026a. Published February 19, 2026.

Google DeepMind. Gemini 3.5 flash: Model card. https://deepmind.google/models/model-cards/gemini-3-5-flash/, May 2026b. Published May 19, 2026.

Aaron Grattafiori et al. The llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

Daya Guo et al. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638, 2025.

Simeng Han et al. 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.

Binyuan Hui et al. Qwen2.5-coder technical report, 2024. URL https://arxiv.org/abs/2409.12186.

Bowen Jiang, Yangxinyu Xie, Zhuoqun Hao, Xiaomeng Wang, Tanwi Mallick, Weijie J Su, Camillo Jose Taylor, and Dan Roth. A peek into token bias: Large language models are not yet genuine reasoners. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 4722–4756, Miami, Florida, USA, November 2024. Association for Computational Linguistics.

Bill Yuchen Lin, Ronan Le Bras, Kyle Richardson, Ashish Sabharwal, Radha Poovendran, Peter Clark, and Yejin Choi. ZebraLogic: On the scaling limits of LLMs for logical reasoning. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 37889–37905. PMLR, 13–19 Jul 2025.

Alexander H. Liu et al. Ministral 3, 2026. URL https://arxiv.org/abs/2601.08584.

Hanmeng Liu, Jian Liu, Leyang Cui, Zhiyang Teng, Nan Duan, Ming Zhou, and Yue Zhang. Logiqa 2.0—an improved dataset for logical reasoning in natural language understanding. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 31:2947–2962, 2023. doi: 10.1109/TASLP.2023.3293046.

Jian Liu, Leyang Cui, Hanmeng Liu, Dandan Huang, Yile Wang, and Yue Zhang. Logiqa: A challenge dataset for machine reading comprehension with logical reasoning. In Christian Bessiere (ed.), *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pp. 3622–3628. International Joint Conferences on Artificial Intelligence Organization, 7 2020. doi: 10.24963/ijcai.2020/501. Main track.

R. Thomas McCoy, Ellie Pavlick, and Tal Linzen. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 3428–3448, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1334.

Aaron Meurer et al. Sympy: symbolic computing in python. *PeerJ Computer Science*, 3:e103, January 2017. ISSN 2376-5992. doi: 10.7717/peerj-cs.103. URL https://doi.org/10.7717/peerj-cs.103.

Aakanksha Naik, Abhilasha Ravichander, Norman Sadeh, Carolyn Rose, and Graham Neubig. Stress test evaluation for natural language inference. In *Proceedings of the 27th International Conference on Computational Linguistics*, pp. 2340–2353, Santa Fe, New Mexico, USA, August 2018. Association for Computational Linguistics.

Mihir Parmar, Nisarg Patel, Neeraj Varshney, Mutsumi Nakamura, Man Luo, Santosh Mashetty, Arindam Mitra, and Chitta Baral. LogicBench: Towards systematic evaluation of logical reasoning ability of large language models. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 13679–13707, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.739.

Nisarg Patel, Mohith Kulkarni, Mihir Parmar, Aashna Budhiraja, Mutsumi Nakamura, Neeraj Varshney, and Chitta Baral. Multi-LogiEval: Towards evaluating multi-step logical reasoning ability of large language models. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 20856–20879, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.1160.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026. URL https://qwen.ai/blog?id=qwen3.5.

Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. Beyond accuracy: Behavioral testing of NLP models with CheckList. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 4902–4912, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.442.

Soumya Sanyal, Zeyi Liao, and Xiang Ren. RobustLR: A diagnostic benchmark for evaluating logical robustness of deductive reasoners. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 9614–9631, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics.

Abulhair Saparov and He He. Language Models Are Greedy Reasoners: A Systematic Formal Analysis of Chain-of-Thought. In *International Conference on Learning Representations*, 2023.

Aaditya Singh et al. Openai gpt-5 system card. *arXiv preprint arXiv:2601.03267*, 2025. doi: 10.48550/arXiv.2601.03267. URL https://arxiv.org/abs/2601.03267.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling llm test-time compute optimally can be more effective than scaling parameters for reasoning. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 10131–10165, 2025.

Alessandro Stolfo, Zhijing Jin, Kumar Shridhar, Bernhard Schölkopf, and Mrinmaya Sachan. A causal framework to quantify the robustness of mathematical reasoning with language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 545–561, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.32.

Oyvind Tafjord, Bhavana Dalvi, and Peter Clark. ProofWriter: Generating implications, proofs, and abductive statements over natural language. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pp. 3621–3634, Online, August 2021. Association for Computational Linguistics.

Ryan Teknium, Roger Jin, Jai Suphavadeeprasit, Dakota Mahan, Jeffrey Quesnelle, Joe Li, Chen Guang, Shannon Sands, and Karan Malhotra. Hermes 4 technical report, 2025. URL https://arxiv.org/abs/2508.18255.

Jidong Tian, Yitian Li, Wenqing Chen, Liqiang Xiao, Hao He, and Yaohui Jin. Diagnosing the first-order logical reasoning ability through LogicNLI. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 3738–3747, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.303.

Siyuan Wang, Zhongkun Liu, Wanjun Zhong, Ming Zhou, Zhongyu Wei, Zhumin Chen, and Nan Duan. From lsat: The progress and challenges of complex reasoning. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 30:2201–2216, 2022. doi: 10.1109/TASLP.2022.3164218.

Zhaofeng Wu, Linlu Qiu, Alexis Ross, Ekin Akyürek, Boyuan Chen, Bailin Wang, Najoung Kim, Jacob Andreas, and Yoon Kim. Reasoning or reciting? exploring the capabilities and limitations of language models through counterfactual tasks. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 1819–1862, Mexico City, Mexico, June 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.naacl-long.102.

An Yang et al. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

William J Youden. Index for rating diagnostic tests. *Cancer*, 3(1):32–35, 1950.

Weihao Yu, Zihang Jiang, Yanfei Dong, and Jiashi Feng. ReClor: A Reading Comprehension Dataset Requiring Logical Reasoning. In *International Conference on Learning Representations*, 2020.

A Dataset Construction Details

A.1 Logical Instance Generation and Label Verification

Logical representation and task semantics. We represent each problem as a logical instance $x = (P, c)$, where $P = \{p_1, \ldots, p_n\}$ is the set of premises and c is the conclusion to be evaluated. Atomic propositions take Boolean truth values, and a *literal* is either an atomic proposition or its negation. The generator constructs formulas from literals using implication, conjunction, and disjunction, while negation is encoded by the polarity of a literal. The corresponding representations and their semantics are summarized in Table 2.

Table 2: Internal representations of propositional-logic formulas used in dataset generation. The table lists propositional-logic formulas used in dataset generation together with their mathematical forms, internal representations, and logical semantics.

Type	Math. form	Internal representation	Semantics
Positive literal	a	`lit(a, pol=True)`	Atom a is true
Negative literal	$\neg a$	`lit(a, pol=False)`	Atom a is false
Implication	$\varphi \to \psi$	`imp(p, q)`	Equivalent to $\neg\varphi \vee \psi$
Conjunction	$\varphi \wedge \psi$	`and(a, b)`	Both subformulas are true
Disjunction	$\varphi \vee \psi$	`or(a, b)`	At least one subformula is true (inclusive disjunction)

The task follows the entailment semantics of propositional logic. An instance is labeled `entailed` if and only if every truth assignment that satisfies all premises also satisfies the conclusion; otherwise, it is labeled `not-entailed`. The latter therefore includes both cases in which the premises support the negation of the conclusion and cases in which the premises are insufficient to determine the conclusion. We additionally require P to be satisfiable, thereby excluding inconsistent premise sets, which would entail arbitrary conclusions. Entities and attributes in the natural-language realization are used only to name atomic propositions, which introduce no additional commonsense knowledge, background axioms, or relations among entities.

Proof-chain construction rules. The generator starts from an initial literal L_0 and incrementally extends a sequence of derivable literals $L_0, L_1, \ldots, L_d$. At each expansion step, the current literal L has already been derived from the premises introduced in earlier steps. The generator then adds the premise required by the selected inference rule and takes a new literal M as the output of that step. For every expansion, it records both the rule identifier and the newly derived literal, yielding a generated proof π together with a rule sequence ρ. We use the depth d to denote the number of proof-chain expansion steps performed by the generator. Table 3 lists all seven construction rules. In the table, L, M, N, X, A, B, and K denote literals. The notation $\neg L$ denotes polarity reversal: if $L = \neg a$, then $\neg L = a$.

Table 3: Inference rules used to construct proof chains. The table lists seven proof-chain construction rules with the available literal, premises added at each step, and derived literal.

ID	Inference rule	Available literal	Premise(s) added at this step	Derived literal
MP	Modus Ponens	L	$L \to M$	M
MT	Modus Tollens	L	$\neg M \to \neg L$	M
DS	Disjunctive Syllogism	L	$\neg L \vee M$	M
HS	Hypothetical Syllogism	L	$L \to X$; $X \to M$	M
CE	Conjunction Elimination	L	$L \to (M \wedge K)$	M
CI	Conjunction Introduction	L	N; $(L \wedge N) \to M$	M
CASES	Proof by Cases	L	$L \to (A \vee B)$; $A \to M$; $B \to M$	M

Table 4: Forms used to generate distractor premises. The table lists single literals, implications, and disjunctions over fresh atoms, with an example of each form and the corresponding condition under which the target entailment relation is preserved.

Form	Example	Why the Target Entailment Relation Is Preserved
Single literal	u or $\neg u$	The fresh atom is independent of the proof chain and the conclusion
Implication	$u \to v$	The constraint involves only newly introduced atoms
Disjunction	$u \vee v$	The constraint involves only newly introduced atoms and remains satisfiable

Table 5: Construction of entailed and not-entailed instances. The table describes the ways used to create entailed, flip, and drop variants from a base instance and their expected label.

Variant	Transformation applied to the positive instance	Expected label
entailed	Retain the proof-supporting premises and the chain-final conclusion c	`entailed`
flip	Retain the premises and replace conclusion c with $\neg c$	`not-entailed`
drop	Remove one proof-chain premise, retain the conclusion, and add a distractor premise over a fresh atom	`not-entailed`

Table 6: Example of label construction for positive, flip, and drop instances. The table provides the premises, conclusion, entailment label, and label justification for each of the three variants of an example proof chain.

Instance	Premises	Conclusion	Label	Justification
Positive	$p,\ p \to q,\ q \to r$	r	`entailed`	Every assignment satisfying the premises also satisfies r
flip	$p,\ p \to q,\ q \to r$	$\neg r$	`not-entailed`	The premises entail r rather than $\neg r$
drop	$p,\ q \to r,\ u$	r	`not-entailed`	After removing $p \to q$, the assignment $p = 1, q = 0, r = 0, u = 1$ satisfies all premises but falsifies r.

Distractor premises. In addition to the premises required by the proof, the generator may insert k irrelevant premises, which serve as distractors. Each distractor is constructed exclusively from fresh atoms that are disjoint from the proof chain and is sampled from one of the forms shown in Table 4.

Pairing entailed and non-entailed instances. Each generated proof skeleton first yields an instance whose conclusion is derivable from the premises and is therefore labeled `entailed`. A paired instance labeled `not-entailed` is then constructed using either the flip or drop operation. This procedure produces equal numbers of positive and negative examples. Every instance is validated using both SymPy and exhaustive truth-table evaluation to ensure that the computed label agrees with the intended label.

Example of label construction. Consider the following two-step modus ponens (MP) chain:

$$P = \{p,\ p \to q,\ q \to r\}, \qquad c = r. \tag{12}$$

From p and $p \to q$, we derive q; from q and $q \to r$, we then derive r. The resulting positive instance is therefore labeled `entailed`. The two corresponding types of negative instances are constructed as shown in Table 6.

A.2 Construction of the Representation Set

Four logically equivalent representations. Each logical instance is rendered in four logically equivalent surface forms, NATURAL, STRIPPED, CODE, and COSMETIC-CODE, while preserving the same underlying premise formulas, conclusion, and label. For the examples in Table 7, the atomic propositions are mapped consistently as follows: p = "Alice is ready," q = "Bob is calm," and r = "Carol is happy."

A.3 Construction of the Intervention Set

The INTERVENTION SET is generated independently from the REPRESENTATION SET, and all base instances are rendered in the STRIPPED format. For each base instance $x = (P, c)$, we construct both logic-changing and logic-preserving transformations.

Table 7: Four logically equivalent representation formats used in the REPRESENTATION SET. The table describes the atomic propositions, expression of logical relations, representational characteristics, and an example for the NATURAL, STRIPPED, CODEL, and COSMETIC-CODE formats.

Representation	Atomic propositions	Logical relations	Characteristics	Example

NATURAL	English declarative statements formed from entity–property pairs	English templates for conditionals, conjunction, disjunction, and negation	Natural language preserving lexical semantics.	`Premises:`
				`1. Alice is ready.`
				`2. If Alice is ready, then Bob is calm.`
				`3. If Bob is calm, then Carol is happy.`
				
				`Claim: Carol is happy.`

STRIPPED	Uppercase proposition symbols	English templates for logical connectives are retained	Entity/property terms replaced by abstract symbols.	`Premises:`
				`1. P is true.`
				`2. If P, then Q.`
				`3. If Q, then R.`
				
				`Claim: R is true.`

CODE	Boolean variables `v0`, `v1`, …	Executable `premises()` and `claim()` functions	Boolean expressions in executable functions.	`def premises(v0, v1, v2):`
				`  facts = [`
				`  v0,`
				`  ((not v0) or v1),`
				`  ((not v1) or v2),`
				`  ]`
				`  return all(facts)`
				
				`def claim(v0, v1, v2):`
				`  return v2`

COSMETIC-CODE	The same Boolean variables	A Python dictionary storing formulas as strings	Code-like syntax and logical formulas without executable functions.	`logic_spec = {`
				`  "premises": [`
				`  "v0",`
				`  "((not v0) or v1)",`
				`  "((not v1) or v2)",`
				`  ],`
				`  "claim": "v2",`
				`}`

Table 8: Logic-changing intervention operators. The table lists conclusion negation, proof-chain premise deletion, literal-polarity flipping, and conjunction/disjunction swapping together with their operator identifiers and implementation details.

ID	Operation	Implementation details
`mc_negate`	Negate the conclusion	Invert the polarity of the conclusion literal while leaving all premises unchanged.
`mc_delete`	Delete a proof-chain premise	Deterministically select and remove one premise marked as part of the proof chain; the seed premise is never removed, and no replacement premise is inserted.
`mc_flip`	Flip literal polarity	Select one literal occurrence in a premise formula and invert its polarity; other occurrences of the same atom are not globally replaced.
`mc_swap`	Swap conjunction and disjunction	Swap conjunction and disjunction nodes occurring in a premise formula while leaving the conclusion unchanged.

Logic-changing and logic-preserving interventions. Logic-changing and logic-preserving interventions are shown in Table 8 and Table 9. The `mc_delete` operator in the INTERVENTION SET differs from the drop operation used in the REPRESENTATION SET. The drop operation inserts a distractor premise so that positive and negative instances have the same number of premises, whereas `mc_delete` directly measures the effect of removing a premise; consequently, the intervened instance contains one fewer premise. Likewise, `mc_flip` should not be confused with symbol renaming. The former changes the logical content of a single literal occurrence within a premise formula, whereas renaming applies a consistent bijection to all occurrences of each atom and thereby preserves the correspondence among formulas.

Table 9: Logic-preserving intervention operators. The table lists irrelevant-premise addition, premise reordering, and consistent symbol renaming together with their operator identifiers and implementation details.

ID	Operation	Implementation details
`lp_neutral`	Add an irrelevant premise	Add a literal, implication, or disjunction constructed from fresh atoms.
`lp_reorder`	Reorder premises	Shuffle the premise list using a deterministic randomized procedure.
`lp_rename`	Rename symbols	Construct a bijection over the atoms in the instance and apply it consistently to all premises and the conclusion.

Table 10: Paired examples for the seven intervention operators. The table shows the transformed premises, conclusion, resulting label, and label justification for four logic-changing interventions and three logic-preserving interventions.

Intervention	Premises	Conclusion	Label	Justification
Conclusion negation	$p,\ p \to q,\ q \to r$	$\neg r$	`not-entailed`	The original premises entail r.
Premise deletion	$p,\ q \to r$	r	`not-entailed`	$p = 1, q = 0, r = 0$ is a counterexample.
Literal-polarity flip	$\neg p,\ p \to q,\ q \to r$	r	`not-entailed`	$p = 0, q = 0, r = 0$ is a counterexample.
Conjunction/disjunction swap	$p,\ p \to (q \vee r)$	q	`not-entailed`	Replacing $q \wedge r$ in the original instance with $q \vee r$ admits $p = 1, q = 0, r = 1$, which satisfies all premises while falsifying q.
Irrelevant-premise addition	$p,\ p \to q,\ q \to r,\ u$	r	`entailed`	The original proof remains valid.
Premise reordering	$q \to r,\ p,\ p \to q$	r	`entailed`	The set of premises is unchanged.
Consistent symbol renaming	$q,\ q \to r,\ r \to p$	p	`entailed`	Applying the bijection $p \mapsto q, q \mapsto r, r \mapsto p$ preserves the proof structure.

Examples of paired interventions. Table 10 illustrates all seven intervention types. Except for the `swap` row, each intervention starts from the base instance $P = \{p, p \to q, q \to r\}$ with $c = r$. For the `swap` row, the base instance is $P = \{p, p \to (q \wedge r)\}$ with $c = q$. Both base instances are labeled `entailed`. The table reports the intervened instances together with their resulting labels.

B Complete Experimental Data on ReliLogic

B.1 Results Across All 74 Configurations

The detaied results of all 74 configurations are shown in Table 11.

B.2 Generalization under Controlled Distribution Shifts and External Benchmarks

The performance differences among models observed in our evaluation generalize across distribution shifts. We evaluate 10 representative model configurations under controlled distribution shifts and on external benchmarks. For the controlled distribution shifts, we vary proof depth, the number of distractors, rule sequences, and lexical realizations. For external evaluation, we use three publicly available datasets: ProofWriter (Tafjord et al., 2021), PrOntoQA (Saparov & He, 2023), and FOLIO (Han et al., 2024).

To quantify the consistency of relative model rankings across different evaluation conditions, we compute Spearman's rank correlation coefficient, ρ, over the same set of models:

$$\rho = \frac{\sum_{i=1}^{n}(R_i - \bar{R})(S_i - \bar{S})}{\sqrt{\sum_{i=1}^{n}(R_i - \bar{R})^2}\sqrt{\sum_{i=1}^{n}(S_i - \bar{S})^2}}, \tag{13}$$

where R_i and S_i denote the ranks of model i under two evaluation conditions, and $\bar{R}$ and $\bar{S}$ denote the corresponding mean ranks.

The Spearman correlations between in-distribution accuracy and controlled-shift accuracy, and between in-distribution accuracy and external-benchmark accuracy, are $\rho = 0.964$ and $\rho = 0.821$,

Table 11: Full results over 74 model configurations. All values except J_I are percentages.

Model	Natural ↑	Stripped ↑	Code ↑	Cosmetic ↑	Overall ↑	ALL-4 ↑	Rep. Dis. ↓	Sens. ↑	Spec. ↑	J_I ↑
Ark-deepseek-r1-0528	91.19	87.74	77.71	75.40	83.01	59.90	36.95	53.49	91.57	0.45
azure-grok-4	99.86	99.88	99.88	99.88	99.88	99.50	0.50	99.71	100.00	1.00
gemini-2.5-flash	98.76	99.48	99.29	99.38	99.23	97.05	2.95	98.57	99.58	0.98
gemini-2.5-flash-lite	97.95	99.52	98.07	96.43	97.99	92.57	7.43	99.29	98.91	0.98
gemini-2.5-pro	99.69	99.93	99.60	99.26	99.62	98.50	1.50	99.14	99.92	0.99
gemini-3-flash-preview	100.00	100.00	99.86	99.98	99.96	99.83	0.17	100.00	100.00	1.00
gemini-3.1-flash-lite	99.81	99.90	95.60	96.90	98.05	92.74	7.26	99.71	99.92	1.00
gemini-3.1-p	99.76	98.95	99.95	99.95	99.65	98.69	1.31	98.57	99.58	0.98
gemini-3.5-flash	95.98	96.38	99.88	99.86	98.03	92.74	7.26	94.97	97.86	0.93
gemini-3.5-flash-lite	99.55	99.93	95.21	97.21	97.98	92.67	7.33	99.86	99.75	1.00
gemini-3.6-flash	100.00	100.00	97.86	99.76	99.41	97.67	2.33	100.00	100.00	1.00
gemini-3.7-flash	100.00	100.00	96.17	99.40	98.89	95.86	4.14	100.00	99.92	1.00
gpt-5-2025-08-07	99.90	99.98	100.00	99.98	99.97	99.86	0.14	100.00	100.00	1.00
gpt-5-mini-2025-08-07	99.02	98.95	99.95	99.98	99.48	97.98	2.02	99.14	99.50	0.99
gpt-5-nano-2025-08-07	98.24	97.02	99.48	98.90	98.41	93.81	6.19	95.33	97.06	0.92
gpt-5.3-codex-2026-02-24	83.57	85.14	76.21	75.38	80.08	55.69	41.38	71.07	91.38	0.62
gpt-5.4-2026-03-05	82.31	82.81	75.88	74.31	78.83	53.67	43.48	70.30	89.15	0.59
gpt-5.4-mini-2026-03-17	77.98	73.02	65.69	70.38	71.77	38.14	57.95	56.09	80.45	0.37
gpt-5.5-2026-04-24	100.00	99.81	99.95	99.98	99.94	99.74	0.26	100.00	100.00	1.00
gpt-5.6-sol	97.71	99.83	99.60	99.88	99.26	97.19	2.81	99.86	100.00	1.00
gpt-oss-120b	98.55	99.93	99.79	99.93	99.55	98.19	1.81	100.00	100.00	1.00
Mistral-Small-4-119B-2603	99.98	99.88	99.38	99.43	99.67	98.69	1.31	99.86	100.00	1.00
NVIDIA-Nemotron-3-Super-120B-A12B-FP8	99.79	99.95	99.93	99.90	99.89	99.57	0.43	100.00	100.00	1.00
Hermes-4-70B	99.62	99.90	96.90	96.36	98.19	93.19	6.81	100.00	99.92	1.00
Qwen3-Next-80B-A3B-Thinking-FP8	99.81	99.36	99.98	99.93	99.77	99.10	0.90	99.71	99.66	0.99
Qwen3.5-122B-A10B-FP8	99.79	99.93	99.74	99.83	99.82	99.29	0.71	100.00	99.92	1.00
Step-3.5-Flash-FP8	99.90	99.98	99.67	99.93	99.87	99.48	0.52	100.00	100.00	1.00
Step-3.7-Flash-FP8	99.76	100.00	99.93	99.90	99.90	99.60	0.40	100.00	100.00	1.00
MiniMax-M2.5	98.88	99.19	99.26	99.02	99.09	96.43	3.57	98.84	99.58	0.98
Qwen3-235B-A22B-Instruct-2507-FP8	82.00	87.52	69.50	71.90	77.73	62.52	29.33	62.65	92.88	0.56
GLM-4.7-FP8	86.64	92.50	75.29	88.52	85.74	60.83	39.07	88.17	92.74	0.81
Qwen3.5-397B-A17B-FP8	99.98	99.98	99.98	99.98	99.98	99.90	0.10	100.00	100.00	1.00
Llama-4-Maverick-17B-128E-Instruct-FP8	94.95	97.45	87.10	92.52	93.01	77.10	22.81	92.32	97.94	0.90
baidu/ernie-4.5-vl-424b-a47b	56.29	63.19	50.07	48.62	54.54	36.26	44.02	17.80	76.78	-0.05
Falcon-H1R 7B	98.93	98.33	98.26	98.62	98.53	94.57	5.43	98.13	98.56	0.97
EXAONE Deep 7.8B	98.64	99.60	96.31	94.17	97.18	90.02	9.98	99.86	99.58	0.99
Llama 3.1 8B Instruct	69.24	62.76	50.86	50.57	58.36	14.21	82.98	56.34	67.60	0.24
Qwen3 8B	98.60	99.95	99.14	99.26	99.24	97.05	2.95	99.71	99.92	1.00
DeepSeek R1 Distill Llama 8B	92.60	95.83	92.64	89.50	92.64	75.10	24.88	95.72	94.49	0.90
Granite 3.3 8B Instruct	70.00	68.00	49.26	41.12	57.09	13.02	84.95	40.28	78.48	0.19
Ministral 3 8B Instruct	70.74	70.64	52.93	59.33	63.41	28.43	63.60	40.79	79.77	0.21
Ministral 3 8B Reasoning	94.57	96.19	82.00	82.38	88.78	64.48	35.45	96.56	96.52	0.93
Nemotron Nano 9B v2	99.17	99.88	98.60	98.83	99.12	96.62	3.38	99.71	99.92	1.00
Gemma 3 12B IT	79.10	75.29	60.26	56.40	67.76	33.24	63.05	51.86	86.09	0.38
Mistral Nemo 12B	69.81	62.43	57.55	56.31	61.52	17.38	78.90	49.49	69.38	0.19
Qwen3 14B	98.95	99.79	99.67	99.64	99.51	98.14	1.86	99.28	99.75	0.99
Qwen2.5 14B Instruct	80.24	75.24	56.62	62.45	68.64	42.60	49.00	60.17	84.14	0.44
Qwen2.5 Coder 14B	77.60	73.45	54.74	55.10	65.22	34.55	58.81	60.21	75.33	0.36
DeepSeek R1 Distill Qwen 14B	97.38	98.43	94.86	95.05	96.43	87.05	12.95	98.27	97.73	0.96
Phi-4 Reasoning	99.26	99.55	99.26	99.21	99.32	97.45	2.55	99.71	99.66	0.99
Phi-4 Reasoning Plus	97.79	96.69	98.98	99.10	98.14	94.14	5.86	94.01	97.93	0.92
Ministral 3 14B Instruct	72.02	73.33	52.29	57.12	63.69	18.81	79.31	49.48	81.11	0.31
Ministral 3 14B Reasoning	97.86	96.64	96.24	96.40	96.78	90.19	9.71	98.06	97.33	0.95
Kimi-VL A3B Thinking	87.26	89.48	74.71	77.57	82.25	48.79	51.05	82.95	87.84	0.71
InternVL3.5 GPT-OSS 20B-A4B Preview	90.71	90.45	77.86	75.48	83.62	56.17	42.86	79.30	89.60	0.69
Mistral Small 3.1 24B	77.00	76.36	57.45	58.14	67.24	27.14	69.14	53.48	80.81	0.34
Devstral Small 2 24B	73.64	72.55	43.93	52.29	60.60	15.00	83.24	58.96	71.90	0.31
Magistral Small 24B	99.52	99.76	99.14	99.17	99.40	97.67	2.33	99.86	99.75	1.00
Gemma 4 26B-A4B IT	80.17	80.36	50.10	50.12	65.19	34.98	59.02	73.54	86.12	0.60
Gemma 3 27B IT	79.36	77.69	52.10	60.07	67.30	37.26	55.90	49.18	87.92	0.37
Qwen3.5 27B	99.33	99.74	97.40	99.07	98.88	95.71	4.29	99.71	99.92	1.00
Qwen3.6 27B	95.62	94.02	95.64	90.43	93.93	81.31	18.55	97.84	98.03	0.96
Qwen3.8 27B	98.86	99.43	99.31	99.12	99.18	96.90	3.10	99.71	99.33	0.99
Muse Glimmer 30B	100.00	99.93	99.86	99.86	99.91	99.64	0.36	99.86	100.00	1.00
Qwen3 30B-A3B Instruct	82.10	80.98	63.40	72.64	74.78	46.88	47.93	50.08	91.48	0.42
Qwen3 30B-A3B Thinking	80.05	85.98	98.19	99.83	91.01	68.69	31.31	82.94	86.86	0.70
InternVL3.5 30B-A3B	74.48	75.83	56.83	58.98	66.53	35.38	56.05	43.33	86.69	0.30
Gemma 4 31B IT	64.55	64.79	50.26	50.19	57.45	49.88	16.50	8.61	97.35	0.06
Qwen3 32B	98.00	99.79	99.43	99.71	99.23	97.00	3.00	100.00	99.66	1.00
QwQ-32B	99.93	99.93	98.62	99.14	99.41	97.71	2.29	99.71	100.00	1.00
DeepSeek R1 Distill Qwen 32B	99.05	98.62	98.79	98.31	98.69	94.95	5.05	98.69	99.32	0.98
Qwen2.5 32B Instruct	82.12	77.98	71.29	66.17	74.39	52.95	39.10	68.71	83.17	0.52
EXAONE 4.0 32B	99.21	99.33	98.62	99.12	99.07	96.38	3.62	99.14	99.83	0.99
OLMo 3.1 32B Think	99.48	99.90	99.71	99.40	99.62	98.55	1.45	100.00	99.83	1.00

respectively. The correlation between controlled-shift accuracy and external-benchmark accuracy is $\rho = 0.906$. These out-of-distribution and external evaluations suggest that the relative performance differences observed on our controlled synthetic benchmark transfer beyond the original in-distribution generation setting, rather than being confined to a particular synthetic data distribution.

Accuracy(%)	ID	Controlled Distribution Shifts					OOD				Ranking
		Depth	Distr.	Held Rules	Lexical	Overall	FOLIO	PrOntoQA	Proof Writer	Overall	
Gemini 3.1 Pro (4096)	99.54	99.78	97.43	99.78	99.49	99.12	89.00	100.00	99.25	96.08	1
Qwen3 8B Thinking ON	99.26	98.03	99.13	97.96	99.26	98.60	86.75	97.75	97.25	93.92	2
GPT-5 Nano	98.30	96.51	97.28	97.62	98.66	97.53	85.75	98.00	100.00	94.58	3
Hermes-4-70B	98.19	95.34	99.47	95.26	98.10	97.05	87.00	98.25	80.50	88.58	4
Gemini 3.5 Flash (medium)	97.93	99.52	94.86	99.25	97.80	97.86	89.75	100.00	98.50	96.08	5
Ministral 3 14B Reasoning	97.17	95.38	94.21	96.92	97.36	95.98	85.50	89.00	92.00	88.83	6
Qwen3 30B-A3B Thinking	91.25	92.12	94.30	90.26	92.43	92.28	85.75	99.25	95.75	93.58	7
Qwen3 30B-A3B Instruct	74.76	67.91	73.32	70.26	77.27	72.24	78.75	67.75	71.50	72.67	8
Qwen3 8B Thinking OFF	72.05	63.27	78.25	64.28	72.18	69.52	74.75	70.25	69.00	71.33	9
Ministral 3 14B Instruct	63.74	58.56	68.10	61.08	65.16	63.24	70.50	63.25	63.25	65.67	10

Figure 8: Performance of representative models under controlled distribution shifts and external benchmarks. The figure reports performance under shifts in proof depth, distractor count, held-out rules, and lexical realizations, together with results on FOLIO (Han et al., 2024), PrOntoQA (Saparov & He, 2023), and ProofWriter (Tafjord et al., 2021) and the corresponding aggregate scores and model ranking.

Table 12: Comparison of reasoning and non-reasoning configurations within the same model families. All values except J_I are percentages.

Model	Configuration	Accuracy	Representation Set		Intervention Set		
		Overall ↑	ALL-4 ↑	Rep. Dis. ↓	Sensitivity ↑	Specificity ↑	J_I ↑
Ministral 3 8B	Instruct	63.41	28.43	63.60	40.79	79.77	0.2056
	Reasoning	**88.79**	**64.48**	**35.45**	**96.56**	**96.52**	**0.9308**
Ministral 3 14B	Instruct	63.69	18.81	79.31	49.48	81.11	0.3058
	Reasoning	**96.79**	**90.19**	**9.71**	**98.06**	**97.33**	**0.9539**
Qwen3 30B-A3B	Instruct	74.78	46.88	47.93	50.08	**91.48**	0.4156
	Thinking	**91.01**	**68.69**	**31.31**	**82.94**	86.86	**0.6980**
Qwen3 8B	Thinking OFF	72.05	39.40	54.82	49.09	88.62	0.3771
	Thinking ON	**99.26**	**97.10**	**2.90**	**99.81**	**99.97**	**0.9978**

B.3 Reasoning Configurations Reshape Logical Behavior

Comparing reasoning and non-reasoning configurations within the same model families (Liu et al., 2026; Yang et al., 2025), we find that enabling reasoning leads to substantial improvements in logical reasoning performance. These gains are not only limited to solving more problems correctly, but also extend to markedly stronger cross-representation reliability and selective causal responsiveness. As shown in Table 12, all four same-family comparisons exhibit a consistent overall pattern: enabling reasoning improves not only final-answer accuracy, but also ALL-4 and J_{I}, with gains of at least 16.23, 21.81, and 28.2%, respectively. The largest improvement is observed for Ministral-3-14B. Switching from the Instruct to the Reasoning configuration increases overall accuracy by 33 % and ALL-4 by 71.38%, while J_{I} rises from 0.306 to 0.954. Taken together, these changes indicate a substantial shift in the model's reasoning profile, from broadly weak performance across the evaluated dimensions to markedly stronger logical reasoning behavior.

B.4 Inference-Time Reasoning Scaling

Token budget scaling. As shown in Fig. 9, we evaluate Gemini 3.1 Pro (Google DeepMind, 2026a) under different reasoning token budgets to examine the effect of scaling test-time compute (Snell et al., 2025). Increasing the budget from 0 to 1,024 improves overall accuracy from 93.12% to 99.60%, ALL-4 from 79.24% to 98.40%, and J_{I} from 0.806 to 0.970. Further increasing the

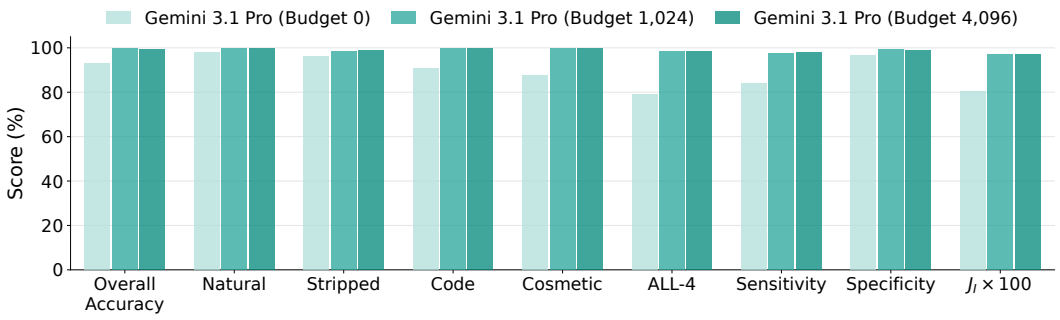

Figure 9: ReliLogic metrics under different reasoning-token budgets of 0, 1,024, and 4,096. The experiment is completed on Gemini 3.1 Pro.

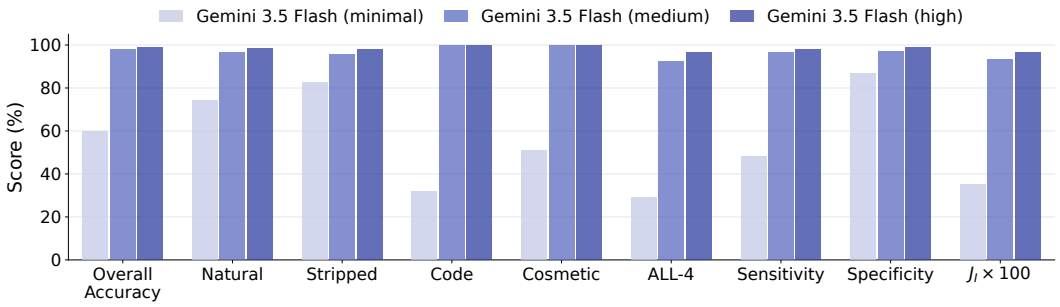

Figure 10: ReliLogic metrics under different reasoning-effort settings of minimal, medium, and high level. The experiment is completed on Gemini 3.5 Flash.

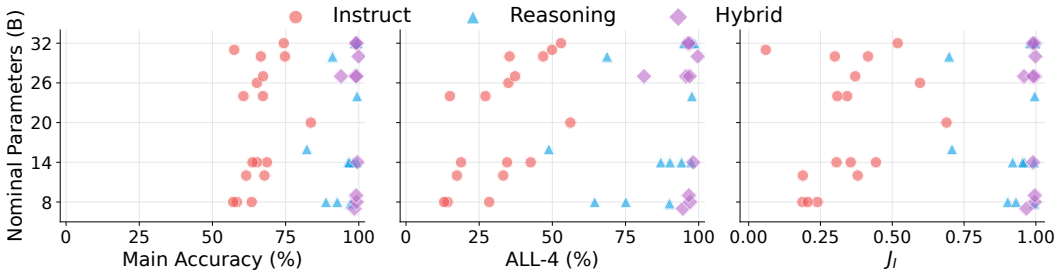

Figure 11: Logical reasoning performance across models of different nominal parameter sizes. The three scatter plots show nominal parameter count against overall accuracy on the REPRESENTATION SET, ALL-4, and J_I, with model configurations categorized as Instruct, Reasoning, or Hybrid.

budget to 4,096 yields corresponding scores of 99.54%, 98.29%, and 0.971, respectively, indicating little additional improvement. Moreover, under the 1,024- and 4,096-token budgets, the model actually consumes an average of approximately 409 and 408 reasoning tokens, respectively. This suggests that increasing the maximum allowable budget does not lead the model to use more reasoning tokens in practice. Overall, allocating a reasoning token budget yields substantial performance gains, whereas further increasing the budget results in diminishing returns, with performance largely saturating beyond 1,024 tokens.

Reasoning effort scaling. As shown in Fig. 10, we evaluate Gemini 3.5 Flash (Google DeepMind, 2026b) under three reasoning-effort settings: *minimal*, *medium*, and *high*. Moving from *minimal* to *medium* substantially improves overall accuracy from 59.93% to 97.93%, ALL-4 from 29.12% to 92.33%, and $J_{\text{intervention}}$ from 0.350 to 0.934. Increasing the reasoning effort further to *high* raises these metrics to 99.05%, 96.45%, and 0.968, respectively. These results show that the largest performance gain occurs when increasing reasoning effort from *minimal* to *medium*. Moving from *medium* to *high* provides further improvements, but with markedly smaller gains, indicating diminishing returns at higher reasoning-effort levels.

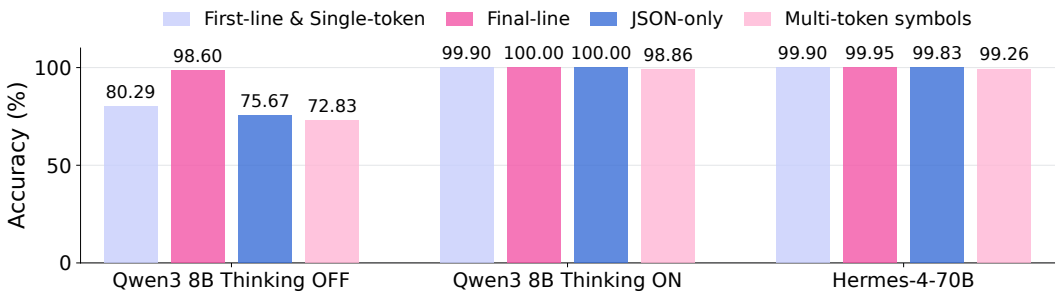

Figure 12: Accuracy under different answer-placement and output-tokenization protocols, including first-line single-token baseline, final-line answers, JSON-only output, and multi-token symbols.

Parameter scaling. We evaluate 40 models ranging from 7B to 32B parameters on a single H100 GPU and examine the relationship between nominal parameter count and logical reasoning performance, as shown in Fig. 11. The Spearman rank correlation between parameter count and overall accuracy is $\rho = 0.276$, indicating only a weak univariate association between model scale and final accuracy within the range considered. More broadly, models with similar parameter counts exhibit substantial variation in performance. Thus, within this model-size regime, a larger parameter count does not necessarily translate into better reasoning performance, suggesting that parameter count alone is a poor predictor of reasoning capability.

B.5 Effects of Answer Placement and Tokenization

We select three model configurations to examine whether differences in model behavior can be attributed to answer placement or the tokenization of output symbols. Under each of four conditions, the models are evaluated on the same 4,200 Stripped logical instances. The conditions include a baseline using a first-line answer with single-token symbols, as well as three variants: placing the answer on the final line, enforcing JSON-only output, and replacing single-token symbols with multi-token symbols. As shown in Fig. 12, Hermes-4-70B (Teknium et al., 2025) and Qwen3 8B Thinking ON (Yang et al., 2025) are largely robust to changes in the answer protocol. In contrast, Qwen3 8B Thinking OFF exhibits substantially greater output-format sensitivity. Requiring the answer to appear on the final line increases its accuracy by 18.31% relative to the baseline, whereas enforcing JSON-only output reduces accuracy by 4.62%. Replacing single-token symbols with multi-token symbols leads to performance degradation across all three configurations, with the largest drop observed for Qwen3 8B Thinking OFF, at 7.45%. These results reveal substantial differences across model configurations in their sensitivity to seemingly superficial interface choices. Moreover, models with weaker reasoning capabilities may also be more vulnerable to variations in reasoning elicitation and tokenization, suggesting that measured reasoning performance can depend not only on the underlying capability of the model but also on how that capability is elicited and how the required outputs are tokenized.