LogicSkills: A Structured Benchmark for Formal Reasoning in Large Language Models

Anonymous ACL submission

Abstract

Large language models have demonstrated notable performance across various logical reasoning benchmarks. However, it remains unclear which core *logical skills* they truly master. To address this, we introduce LogicSkills, a unified benchmark designed to isolate three fundamental skills in formal reasoning: (i) *formal symbolization*—translating premises into first-order logic; (ii) *countermodel construction*—formulating a finite structure in which all premises are true while the conclusion is false; and (iii) *validity assessment*—deciding whether a conclusion follows from a given set of premises. Items are drawn from the two-variable fragment of first-order logic (without identity) and are presented in both natural English and a Carroll-style language with nonce words. All examples are verified for correctness and non-triviality using the SMT solver Z3. Across leading models, performance is high on validity but substantially lower on symbolization and countermodel construction, suggesting reliance on surface-level patterns rather than genuine symbolic or rule-based reasoning.

1 Introduction

Logic, "the science of reasoning," is not psychology's project of describing how we *do* reason (Byrne et al., 1993; Rips, 1994) but of how we *ought* to reason (Frege, 1897/1979). It holds that rational agents should infer what their beliefs entail and avoid inconsistency (Harman, 1986; Field, 2009). In artificial intelligence, reasoning is typically evaluated *behaviorally*—by whether systems succeed on tasks humans typically solve using explicit reasoning—without assuming any particular internal method. In this setting, logic exercises serve as external *proxies* for deductive competence: success should reflect sensitivity to logical structure rather than to surface cues or world knowledge. However, standard benchmarks for logical competence (Tafjord et al., 2021; Parmar et al., 2024) often conflate distinct logical subskills, making it difficult to determine which aspects of reasoning models genuinely master.

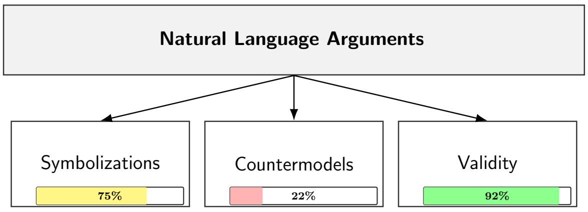

Figure 1: Accuracy of closed models across three skills.

For instance, a model might correctly carry out a formal proof when given symbolic premises (as tested in ProofWriter (Tafjord et al., 2021) or LogicBench (Parmar et al., 2024)), yet struggle to translate natural-language sentences into the symbolic premises. Likewise, a model might apply modus ponens or other inference rules correctly but falter when asked to find a countermodel that demonstrates the invalidity of a step, revealing weaknesses in its model-theoretic understanding.

To address this issue, we introduce LogicSkills, a benchmark for formal reasoning that disentangles fundamental logical skills and exposes failure modes that composite tasks may hide. Drawing on formal logic pedagogy, LogicSkills isolates three fundamental logical skills: (i) *formal symbolization* (mapping sentences to logical form), (ii) *countermodel construction* (demonstrating invalidity via model-theoretic falsification), and (iii) *validity assessment* (recognizing what follows from what). LogicSkills is solver-verified (via Z3 (de Moura and Bjørner, 2008)) and includes a bilingual task design (English and Carrollian), with the Carrollian nonce-word condition isolating reasoning from prior semantic knowledge.

Systematic evaluations on LogicSkills across frontier open-weight and closed-weight large language models (LLMs) reveal a consistent profile: strong performance on validity assessment, but substantial weaknesses in symbolization and countermodel construction (see Figure 1). Together, these results suggest that various LLMs achieve high apparent reasoning accuracy through pattern-based approximations that fall short of the symbolic or rule-based procedures traditionally used in logic.

2 Related Work

Prior evaluations of LLM reasoning divide broadly into two traditions. *Puzzle-based benchmarks* (e.g., knights-and-knaves, logic-grid, and minesweeper tasks) probe combinatorial reasoning often revealing brittleness under search and small perturbations (Mondorf and Plank, 2024; Lin et al., 2025; Giadikiaroglou et al., 2024; Li et al., 2024). *Rule- or proof-oriented suites* such as PROOFWRITER, LOGICBENCH, and LOGICASKER assess inference templates across propositional and first-order logic (Tafjord et al., 2021; Parmar et al., 2024; Han et al., 2024; Wan et al., 2024), but do not disentangle the subskills they presuppose.

LOGICSKILLS complements both traditions by offering a structured, skill-level evaluation framework rather than a single composite reasoning task. The following section presents an overview of the benchmark and details each task type.

3 Benchmark Overview

We build a compositional generator that produces grammatical natural-language sentences paired with formulas in the two-variable fragment of first-order logic (FO^2, without identity). We choose FO^2 since this logic retains decidability and supports finite satisfiability while being expressive enough to capture nontrivial reasoning (Scott, 1962; Mortimer, 1975). Every formula is realized twice: (i) controlled English and (ii) a Carroll-style nonsense lexicon (Carroll, 1871). The latter provides a structurally identical surface form that removes real-world associations and isolates pure logical competence. For example, both the following language examples, i.e., “A human chased a donkey that chased it” and “A borogove snicker-snacked a tove that snicker-snacked it” map to the formula:

$$\exists x \exists y \big((Nx \land My) \land (Rxy \land Ryx) \big) \tag{1}$$

By varying quantifier structure, predicate choice, name assignments, connectives, and recursive clausal patterns, the generator can produce indefinitely many grammatical sentences in both English and Carrollian.

Because every sentence is paired with an FO^2 formula, we leverage SMT solvers (Davis and Putnam, 1960)—in particular the state-of-the-art theorem prover Z3 (de Moura and Bjørner, 2008)—to compute core semantic properties: (i) whether two formulas are equivalent, (ii) whether a set of formulas is jointly satisfiable, (iii) whether one formula follows from others, and (iv) whether a proposed finite model satisfies (or falsifies) a given argument. These capabilities enable large-scale, correctness-guaranteed construction and verification of all deductive tasks in the benchmark. In this way, Z3 is essential in generating arguments. To construct arguments, we sample sets of 3–5 jointly satisfiable premises and use Z3 to search for semantically relevant, non-trivial conclusions that follow from them (valid). For each valid conclusion, we additionally generate 5 structurally similar distractor conclusions—by matching sentence type and symbol set—that do not follow (invalid). Appendix B provides further details about the dataset and task construction pipeline, including sentence generation, argument filtering, and task instantiation.

3.1 Task types

We instantiate three task types, each aligned with a canonical logical skill (Kalish and Montague, 1964). Exact task prompts can be found in Appendix E.

Formal symbolization Given a sentence in English/Carroll and a fixed symbol key, models must output a single well-formed formula capturing the sentence using only the provided predicates, constants, and standard logical operators (i.e., $\neg, \land, \lor, \to, \leftrightarrow, \forall, \exists$).

Input:
Sentence: “All raths are uffish.”
Key: F: “x is a rath”, G: “x is uffish”.

Output: $\forall x\,(Fx \to Gx)$

Countermodel construction For an invalid argument, models must provide a finite structure (on a fixed small domain) that makes all premises true and the conclusion false.

Input:
Invalid Argument: $\forall x\,(Fx \to Gx),\ Ga \vDash Fa.$
Domain: $\{0, 1\}$

Output: $F = \{0\}, G = \{0, 1\}, a = 1.$

Argument validity Given premises and six candidate conclusions in English/Carroll, models must decide which conclusion(s), if any, *must* follow.

Input:

Premises: “All raths are uffish”; “Zindle is a rath.”
Candidate Conclusions:

1. Zindle is not uffish.
2. If Zindle is uffish, then Zindle is a rath.
3. Zindle is uffish.

Output: 3

4 Evaluation Setup

LogicSkills comprises a fixed evaluation set of 1500 problems randomly sampled from generated sentences and arguments: 600 *symbolization* problems (300 English, 300 Carroll), 600 *validity* problems (300 English, 300 Carroll), and 300 *countermodel* problems. Validity assessment and formal symbolization are evaluated bilingually (English and Carroll), whereas countermodel construction is language-neutral.

We evaluate a range of open- and closed-weight LLMs from multiple vendors and across various sizes: *Llama-3.1-8B*, *Llama-3.1-70B*, *Qwen2.5-Math-72B*, *Qwen3-32B*, *Claude-3.7-Sonnet*, *GPT-4o*, *Gemini-2.5-Flash*, *DeepSeek-Chat*, and *Phi-4*. API-based models (OpenAI, Anthropic, Google, DeepSeek) were accessed between July and August 2025. Model responses are processed in two stages. First, an extractor LLM (GPT-4o) cleans and normalizes raw completions.[1] Second, task-specific verification procedures check correctness:

Symbolization Normalized formulas are first checked for an exact match; if necessary, they are parsed and repaired. Cosmetic variants (e.g., $\wedge$ as &, $\neg$ as $\sim$, $\rightarrow$ as $\supset$, or $R(x, y)$ vs. Rxy) are handled via the extractor LLM. Final parses are validated for logical equivalence using Z3.

Countermodels Candidate models are normalized into domain, constants, and predicate extensions. The system enforces *symbol completeness* and validates type/domain consistency (integer domains, correct arities). Candidate models are then translated into SMT-LIB, merged with the negated argument, and checked for satisfiability with Z3.

Validity Extracted labels are compared directly against the gold annotations.

5 Results

Figure 2 illustrates the models’ accuracy across the three different task types described in Section 3. The dominant pattern is a strong *validity* performance contrasted with weaker *symbolization* and especially *countermodel* performance. For instance, GPT-4o achieves 87.0% on validity but only 10.0% on countermodels; Gemini-2.5-Flash reaches 96.0% vs. 13.0%, respectively. The main exception is *Qwen3-32B*, which performs strongly across all three skills (97.0% validity, 85.0% symbolization, 89.0% countermodels).

This striking deviation appears correlated with Qwen3-32B’s inference strategy. Whereas most models rely on generic “reasoning rhetoric” (e.g., “Let’s break this down step-by-step”), Qwen3-32B consistently engages in spontaneous self-scaffolding, marked by explicit internal delimiters (e.g., `<think>`). Its chain-of-thought traces are, on average, $7.5\times$ longer than those of its peers, coinciding with its superior performance across all three tasks.[2] Per-model, per-task accuracies (n=300 per condition) are shown in Table 3 in the Appendix.

Across top-tier models, *validity* accuracy clusters near ceiling, suggesting that premise–conclusion classification is largely mastered under the given conditions. In contrast, *symbolization* remains a consistent bottleneck, reflecting persistent difficulty in mapping natural-language statements to formal representations even when inference competence is high. The *countermodel* task is the most discriminative, revealing substantial gaps in model-theoretic reasoning—with all but Qwen3 failing to generate consistent falsifying models.

5.1 Bilingual evaluation

Perhaps surprisingly, bilingual evaluation shows only minor differences: averaging across models, English and Carroll results are closely aligned (validity 81.1% vs. 82.8%; symbolization 68.1% vs. 66.0%), with a maximum difference of 7 percentage points across models. This null result implies that the models are not relying on semantic heuristics to process the syntactic structure. For an overview of accuracy values per model, task and language, we refer to Table 3 in the Appendix.

---

[1] A meta-evaluation shows 98–99% extraction accuracy; see Appendix D for more details.

[2] Notably, this scaffolding emerges uniformly across all task types. See (Yang et al., 2025) for details on the model’s “thinking mode” architecture and large-scale reasoning training regime.

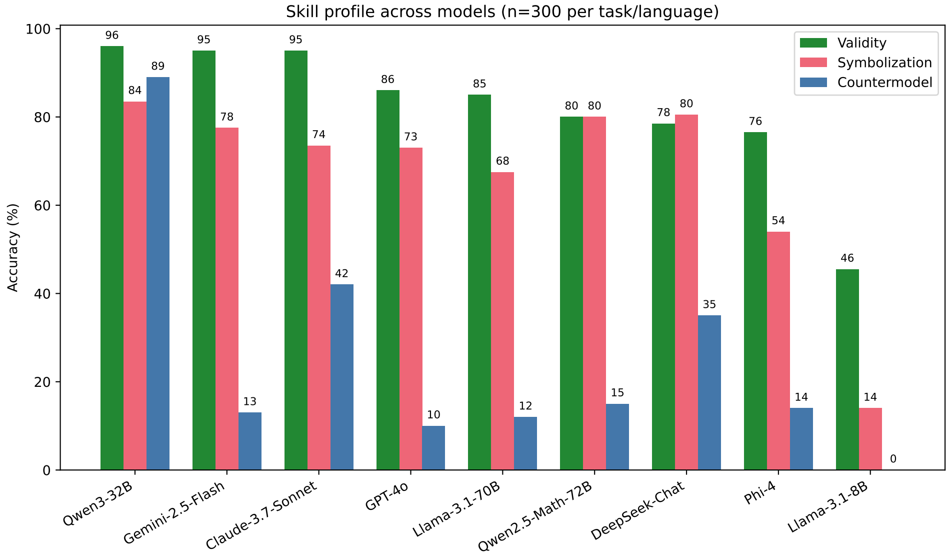

Figure 2: **Skill profile across models** (accuracy %; n=300 per task/language). Grouped bars show *Validity*, *Symbolization*, and *Countermodel*. Most models score highly on validity but lag on subskills, especially countermodels.

5.2 Error analysis

Error analyses across tasks reveal distinct but converging weaknesses in model reasoning. In the *symbolization* task, nearly half of the 1,775 incorrect outputs (46.9%) are semantically wrong despite syntactic well-formedness, while the remainder show structural issues such as predicate-arity mismatches or ill-formed parentheses. In the *countermodel* task, most erroneous outputs (72.7% of 2,010) are structurally well-formed but fail to falsify the argument—typically because one or more premises are false—while the rest (27.3%) reflect missing interpretations or malformed predicate definitions. In the *validity* task, a majority of the 973 incorrect outputs (57.6%) include the correct answer alongside additional invalid options, suggesting difficulty in eliminating plausible distractors. Further details about the error analysis procedure can be found in Appendix C.

5.3 Subskill Transfer via PEFT

In principle, skill at symbolization and countermodel construction should reinforce validity assessment: to judge validity, a model must map sentences to their logical form and evaluate whether any counterexample structure can satisfy the premises while falsifying the conclusion. To assess potential cross-skill transfer, we conduct a fine-tuning study. We fine-tune *Llama-3.2-3B-Instruct* with LoRA adapters on 100k *symbolization* and 100k *countermodel* examples (targets generated via tpg; Schwarz, 2025), using a single epoch with AdamW and standard regularization. After fine-tuning, we observe large *in-task* gains but no measurable transfer to the validity assessment task. Yet, joint fine-tuning on symbolization and countermodel data yields larger countermodel gains than countermodel-only training (+25.6 vs. +15.7), indicating task interaction. Table 1 summarizes the corresponding transfer matrix.

Train Data	**Symb. Δ**	**CM Δ**	**Val. Δ**
Symbolization (100k)	+80.7	0.0	–4.3
Countermodel (100k)	–3.0	+15.7	–6.4
Symb + Counter (200k)	+84.0	+25.6	–5.7

Table 1: Transfer deltas (points vs. base).

6 Conclusion

From Leibniz's (1685) call to "let us calculate" to Frege's *Begriffsschrift* (1879), the project of logic has rested on the conviction that reasoning becomes reliable only when translated into a formal system—where errors can be detected and validity established by explicit rules. Taken together, the results presented in this study suggest that LLMs approximate reasoning success without the explicit symbolic and model-based procedures that underpin the standard methods of logic. Arguably, a system without such explicit control structures cannot satisfy the normative standards that make reasoning accountable and corrigible.

Limitations

While our study introduces a novel benchmark for formal reasoning that disentangles core logical skills and exposes failure modes that composite tasks may hide, there are several limitations that future work can address.

Logical and linguistic scope LOGICSKILLS evaluates reasoning within a deliberately restricted logical and linguistic scope. All tasks are drawn from the two-variable fragment of first-order logic without identity, excluding richer logical resources such as full first-order logic, identity, or modal operators. Consequently, results may not generalize to reasoning in more expressive logical systems. The benchmark also relies on a controlled fragment of English and a synthetically generated Carrollian language. While this design isolates logical structure, it limits linguistic diversity and does not capture many sources of difficulty present in naturally occurring language.

Evaluation procedure Model performance is additionally sensitive to prompting and evaluation protocols. In particular, the countermodel and symbolization tasks require fully specified structured outputs, so measured performance may reflect both reasoning ability and success in satisfying interface and formatting constraints. We partially mitigate this by using liberal extraction and normalization procedures that permit notational variants and apply automated repairs, though some sensitivity remains. Finally, LOGICSKILLS provides a behavioral evaluation only. While it reveals systematic differences across logical subskills, it does not directly address the internal representations or mechanisms by which models arrive at their answers (Chalmers, 2025; Stolfo et al., 2023).

Within the scope of this paper, these limitations are largely intentional design choices, reflecting a tradeoff between expressivity and reliable, solver-verifiable evaluation.

References

Ruth MJ Byrne, Jonathan St BT Evans, and Stephen E Newstead. 1993. *Human reasoning: The psychology of deduction*. Psychology Press.

Lewis Carroll. 1871. *Through the Looking-Glass, and What Alice Found There*. Macmillan and Co., London. Chapter 1, pp. 21–24.

David J. Chalmers. 2025. Propositional interpretability in artificial intelligence. *Preprint*, arXiv:2501.15740.

Martin Davis and Hilary Putnam. 1960. A computing procedure for quantification theory. *J. ACM*, 7(3):201–215.

Leonardo de Moura and Nikolaj Bjørner. 2008. Z3: An efficient smt solver. In *Tools and Algorithms for the Construction and Analysis of Systems*, pages 337–340, Berlin, Heidelberg. Springer Berlin Heidelberg.

Hartry Field. 2009. What is the normative role of logic? *Aristotelian Society Supplementary Volume*, 83(1):251–268.

Gottlob Frege. 1879. *Begriffsschrift: eine der arithmetischen nachgebildete Formelsprache des reinen Denkens*. Verlag von Louis Nebert, Halle. English translation in Jean van Heijenoort (ed.), *From Frege to Gödel: A Source Book in Mathematical Logic, 1879–1931*, Harvard University Press, 1967, pp. 1–82.

Gottlob Frege. 1897/1979. Logic. In Hans Hermes, Friedrich Kambartel, and Friedrich Kaulbach, editors, *Posthumous Writings*, pages 126–151. University of Chicago Press, Chicago. Originally written in 1897.

Panagiotis Giadikiaroglou, Maria Lymperaiou, Giorgos Filandrianos, and Giorgos Stamou. 2024. Puzzle solving using reasoning of large language models: A survey. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 11574–11591, Miami, Florida, USA. Association for Computational Linguistics.

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, and 16 others. 2024. FOLIO: Natural language reasoning with first-order logic. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 22017–22031, Miami, Florida, USA. Association for Computational Linguistics.

Gilbert Harman. 1986. *Change in View: Principles of Reasoning*. MIT Press, Cambridge, MA, USA.

Donald Kalish and Richard Montague. 1964. *Logic: Techniques of Formal Reasoning*. Oxford University Press.

Gottfried Wilhelm Leibniz. 1685. Generales inquisitiones de analysi notionum et veritatum. English translation in Leroy E. Loemker (ed.), *Philosophical Papers and Letters*, 2nd ed., D. Reidel, 1969, pp. 321–325.

Yinghao Li, Haorui Wang, and Chao Zhang. 2024. Assessing logical puzzle solving in large language models: Insights from a minesweeper case study. In *Proceedings of the 2024 Conference of the North*

American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), page 59–81. Association for Computational Linguistics.

Bill Yuchen Lin, Ronan Le Bras, Kyle Richardson, Ashish Sabharwal, Radha Poovendran, Peter Clark, and Yejin Choi. 2025. Zebralogic: On the scaling limits of LLMs for logical reasoning. In *Forty-second International Conference on Machine Learning*.

Philipp Mondorf and Barbara Plank. 2024. Liar, liar, logical mire: A benchmark for suppositional reasoning in large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 7114–7137, Miami, Florida, USA. Association for Computational Linguistics.

Michael Mortimer. 1975. On languages with two variables. *Mathematical Logic Quarterly*, 21(1):135–140.

Mihir Parmar, Nisarg Patel, Neeraj Varshney, Mutsumi Nakamura, Man Luo, Santosh Mashetty, Arindam Mitra, and Chitta Baral. 2024. LogicBench: Towards systematic evaluation of logical reasoning ability of large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13679–13707, Bangkok, Thailand. Association for Computational Linguistics.

Lance J. Rips. 1994. *The Psychology of Proof: Deductive Reasoning in Human Thinking*. The MIT Press.

Wolfgang Schwarz. 2025. Tree proof generator (tpg). `https://github.com/wo/tpg`. See also `https://www.umsu.de/trees/`.

Dana Scott. 1962. A decision method for validity of sentences in two variables. *Journal of Symbolic Logic*, 27(377):74.

Alessandro Stolfo, Yonatan Belinkov, and Mrinmaya Sachan. 2023. A mechanistic interpretation of arithmetic reasoning in language models using causal mediation analysis. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*.

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

Yuxuan Wan, Wenxuan Wang, Yiliu Yang, Youliang Yuan, Jen-tse Huang, Pinjia He, Wenxiang Jiao, and Michael Lyu. 2024. Logicasker: Evaluating and improving the logical reasoning ability of large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 2124–2155, Miami, Florida, USA. Association for Computational Linguistics.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

A Lexicon and Sentence Types

LOGICSKILLS sentences span a range of first-order constructions. Table 2 gives representative examples, covering atomic predicates, quantification (including nested and anaphoric forms), Boolean connectives, and conditionals. Each schema admits systematic surface variants (e.g., VP ellipsis, contrastive conjunctions, unless-disjunctions) that preserve the same logical structure.

Each sentence is realized in two surface languages: controlled English and a Carroll-style nonsense language, each with its own fixed lexicon. The fixed lexicons contain 3 constants (names) and 13 predicates, which bounds the maximum vocabulary size of any sentence.

B Dataset and Task Construction Pipeline

Sentence generation. All tasks are grounded in a large shared sentence inventory (100k+ sentences) generated compositionally. Each sentence is paired with a surface realization (English or Carrollian), an FO^2 logical form, its abstract syntax tree (AST), and a schema of abbreviation (SOA) mapping logical symbols to lexical items. English and Carroll realizations are linked by a counterpart relation preserving logical form. Sentences are labeled by syntactic type (atomic, negated, conditional, quantified, etc.) and stored in a database with fields for surface form, SOA, FO^2 formula, AST, language tag, and a counterpart identifier linking English and Carroll realizations.

Argument generation. Arguments are generated by sampling small premise sets (3–5 sentences) from the sentence table and applying solver-based filtering. Premises must be jointly satisfiable, and candidate conclusions are tested for entailment using Z3. To enforce non-triviality, we exclude theorems, conclusions derivable from domain constraints alone, and conclusions derivable from only a proper subset (at most 20%) of the premises. For each valid argument, five invalid arguments are constructed using alternative conclusions that fail to follow; distractors are constrained to share subject matter with the valid conclusion (same predicates and constants), preferentially match its syntactic type, and be solver-verified as non-entailed. Arguments are stored in a database with fields for premise identifiers, conclusion identifier, validity label, and language, and form the large shared pool from which validity and countermodel tasks are derived.

Task instantiation. Three task families are derived from the sentence and argument tables. Formal symbolization tasks are constructed from individual sentences paired with their SOA, with the associated FO^2 formula as the target. Although the fixed lexicons permit a larger vocabulary in principle, in practice sentences used to instantiate tasks involve only 3–4 predicates and 1–2 constants (4–6 symbols total). Argument validity tasks are constructed from solver-certified valid arguments, each paired with five invalid alternatives sharing the same premises. Countermodel construction tasks are constructed from solver-certified invalid arguments and require a finite model that satisfies all premises while falsifying the conclusion.

C Error Analysis Methodology

Our evaluation pipeline produced structured outputs beyond binary correctness labels, enabling systematic error analysis across all tasks. For each instance, we retained the raw model response, the extracted or parsed answer, the gold-standard answer, and task-specific diagnostic metadata from the evaluator.

Formal symbolization For *symbolization*, each response included an assessment label (e.g., *Not logically equivalent*, *Failed to parse*). Incorrect outputs were filtered and categorized by matching these labels, allowing semantic errors to be distinguished from syntactic or structural failures.

Countermodel construction For *countermodel construction*, the model checker returned a list of diagnostic error messages (e.g., *Not a countermodel*, *Missing interpretation for: X*, *Binary predicate 'P' must be a list of pairs*, etc.). Failures were categorized by matching error-message patterns, with additional checks performed by evaluating premises and conclusions against the proposed structure when needed.

Argument validity For *validity*, both the extracted and correct answers were represented as sets of statement indices. Set comparisons were used to distinguish incorrect selections from *superset* errors in which the correct answer was included alongside additional invalid options.

All error analyses were performed programmatically over the full evaluation set, and the resulting error-type distributions correspond to those reported in the main text.

D Meta-Evaluation of Answer Extraction

Raw model responses were post-processed by an extractor LLM to normalize outputs into task-specific formats prior to scoring. To assess the reliability of this extraction step independently of task correctness, we conducted a meta-evaluation.

We randomly sampled 100 evaluation instances, balanced across all task types (symbolization, validity, and countermodel construction) and languages where applicable, and including both correct and incorrect model responses. For each instance, the evaluator was given the original task prompt, the model's raw response, and the extracted answer produced by the extractor.

The meta-evaluator (GPT-4o) rendered a binary judgment—faithful or unfaithful—indicating whether the extracted answer accurately reflected the content of the raw response, independent of correctness. Five cases were flagged as potentially unfaithful.

All flagged cases, together with a random sample of 10 unflagged instances, were subsequently reviewed by a human evaluator (a meta-meta-evaluation). Of the five flagged cases, one constituted a clear extraction error, in which the model revised a predicate assignment mid-response and the extractor failed to capture the update. One case involved a normalization failure, where a malformed but correctly extracted formula was not repaired. The remaining three cases involved internally inconsistent raw responses (e.g., conflicting predicate arities or self-contradictory conclusions); in these cases, the extractor's output was judged reasonable given the ambiguity of the source response.

After human adjudication, only one instance was deemed a clear extraction failure, with one additional case judged ambiguous, corresponding to an effective extraction accuracy of approximately 98–99%.

E Task Prompts

Figures 3 to 5 illustrate the prompts used for each task type described in Section 3.1.

F Additional Results

Table 3 reports the complete per-model results underlying §5.

G Use of AI Assistants

We used GitHub Copilot for parts of the project's source code and ChatGPT to correct minor grammatical errors.

ARGUMENT VALIDITY PROMPT

Your task is to solve a logical reasoning problem. Use any approach you find effective, but clearly and explicitly state your final answer.

Task

Consider the following situation:

> Everything is a tove, a borogove, or a rath (exclusively), and there’s at least one of each. Zindle or Bungo will whiffle. Only toves will whiffle. Every rath chortled at Bungo. If Zindle will whiffle, then every rath is mimsy only if every borogove chortled at Bungo.

Which, if any, of the following statements must be true in this situation?

1. Zindle and Bungo will whiffle.
2. Every tove will whiffle.
3. Not all toves will whiffle.
4. A tove will whiffle.
5. No toves will whiffle.
6. Zindle will whiffle.

Figure 3: Illustrative instance of the *Argument Validity* task.

FORMAL SYMBOLIZATION PROMPT

Your task is to translate the provided sentence into formal predicate logic, using the abbreviations provided.

Instructions

- Use only the abbreviations given.
- Return a single well-formed formula in standard predicate logic syntax.
- Use standard logical symbols:
  - Quantifiers: $\forall$, $\exists$
  - Connectives: $\neg$, $\wedge$, $\vee$, $\rightarrow$, $\leftrightarrow$
  - Do not include any explanation or extra text—just the formula.

Example

Sentence: Every linguist admires Charlie.
Abbreviations:

- L: “[1] is a linguist”
- R: “[1] admires [2]”
- c: “Charlie”

Translation: $\forall x(Lx \rightarrow Rxc)$

Task

Sentence: Not all raths will whiffle and no toves chortled at Bungo.

Abbreviations:

- O: “[1] is a rath”
- F: “[1] will whiffle”
- M: “[1] is a tove”
- P: “[1] chortled at [2]”
- c: “Bungo”

Figure 4: Illustrative instance of the *Formal Symbolization* task.

COUNTERMODEL CONSTRUCTION PROMPT

Show that the provided argument is invalid by giving a countermodel—one where all premises are true and the conclusion is false.

Instructions

1. Provide assignments for all constants and predicates used in the argument.
2. Respect predicate arity:
    - Monadic predicates take one argument (e.g., Mx).
    - Binary predicates take two arguments (e.g., Pxy).
3. Use the fixed domain $[0, 1, 2]$.

Required Format

- Domain: $[0, 1, 2]$
- Constants: “a”: 0
- Monadic predicates: “F”: $[0, 2]$
- Binary predicates: “R”: $[[0, 1], [2, 0]]$

Argument

$$\big(\forall x\,(Mx \lor (Nx \lor Ox)) \;\land\; \big(\neg\exists x\,(Mx \land Nx) \;\land\; (\neg\exists x\,(Mx \land Ox) \;\land\; \neg\exists x\,(Nx \land Ox))\big)\big),$$

$$(\exists x\,Mx \;\land\; (\exists x\,Nx \;\land\; \exists x\,Ox)),$$

$$\neg\exists x\exists y\,((Mx \land My) \land Pxy),$$

$$(\neg\exists x\,(Ox \land Qxc) \to Mb),$$

$$(\neg\exists x\,(Nx \land Rxc) \to \exists x\,(Nx \land \neg Rxb)),$$

$$(\forall x\,(Ox \to Rxc) \to (Mb \to \forall x\,(Nx \to Pxc))),$$

$$(Pcc \land \forall x\,(Ox \to Rxa))$$

$$\vDash Mc$$

Figure 5: Illustrative instance of the *Countermodel Construction* task.

Sentence type	**English example**	**Carroll example**
atomic_monadic	Hazel drank.	Bungo gyred.
atomic_dyadic	Hazel kicked Lewis.	Bungo galumphed over Rafin.
negation	Lewis isn't happy.	Rafin isn't uffish.
quantified_universal_affirmative	Every monkey is asleep.	Every rath is mimsy.
quantified_particular_affirmative	A donkey will run.	A tove will whiffle.
quantified_universal_negative	No humans chased Alfred.	No borogoves snicker-snacked Zindle.
quantified_particular_negative	Not all donkeys chased Alfred.	Not all toves snicker-snacked Zindle.
quantified_only_restrictor	Only humans will run.	Only borogoves will whiffle.
quantified_name_all	Hazel chased every human.	Bungo snicker-snacked every borogove.
quantified_name_some	Alfred chased a donkey.	Zindle snicker-snacked a tove.
quantified_all_all	Every monkey kicked every donkey.	Every rath galumphed over every tove.
quantified_all_all_all	Every donkey saw every donkey that saw every monkey.	Every tove chortled at every tove that chortled at every rath.
quantified_all_all_back	Every donkey chased every monkey that chased it.	Every tove snicker-snacked every rath that snicker-snacked it.
quantified_all_some	Every monkey kicked a human.	Every rath galumphed over a borogove.
quantified_all_some_back	Every human saw a donkey that saw it.	Every borogove chortled at a tove that chortled at it.
quantified_some_all	A monkey chased every human.	A rath snicker-snacked every borogove.
quantified_some_all_back	A monkey saw every human that saw it.	A rath chortled at every borogove that chortled at it.
quantified_some_some	A human chased a donkey.	A borogove snicker-snacked a tove.
quantified_some_some_back	A donkey kicked a monkey that kicked it.	A tove galumphed over a rath that galumphed over it.
quantified_some_some_some	A human kicked a human that kicked a monkey.	A borogove galumphed over a borogove that galumphed over a rath.
quantified_no_all	No humans kicked every human.	No borogoves galumphed over every borogove.
quantified_no_some	No humans chased a human.	No borogoves snicker-snacked a borogove.
quantified_no_some_back	No donkeys kicked a monkey that kicked it.	No toves galumphed over a rath that galumphed over it.
quantified_rev_some_all	There is a donkey that every monkey kicked.	There is a tove that every rath galumphed over.
quantified_rev_no_all	There is not a monkey that every human chased.	There is not a rath that every borogove snicker-snacked.
quantified_some_self	A donkey kicked itself.	A tove galumphed over itself.
conjunction_simple	Hazel saw Lewis and not all monkeys kicked Lewis.	Bungo chortled at Rafin and not all raths galumphed over Rafin.
disjunction_simple	Alfred drank or a monkey kicked Alfred.	Zindle gyred or a rath galumphed over Zindle.
conditional_if_then	If Hazel is a human, then Alfred isn't a human.	If Bungo is a borogove, then Zindle isn't a borogove.
biconditional_just_in_case	Hazel is happy just in case every donkey chased Lewis.	Bungo is uffish just in case every tove snicker-snacked Rafin.

Table 2: Sentence types used in LogicSkills with representative English and Carroll examples.

Model	Task	Language	Total	Correct	Accuracy
meta_llama-3.1-8B-instruct	Validity	Carroll	300	138	0.46
	Validity	English	300	135	0.45
	Symbolization	Carroll	300	42	0.14
	Symbolization	English	300	41	0.14
	Countermodel	–	300	0	0.00
meta_llama-3.1-70B-instruct	Validity	Carroll	300	251	0.84
	Validity	English	300	257	0.86
	Symbolization	Carroll	300	198	0.66
	Symbolization	English	300	206	0.69
	Countermodel	–	300	36	0.12
qwen2.5-math-72B-instruct	Validity	Carroll	300	241	0.80
	Validity	English	300	239	0.80
	Symbolization	Carroll	300	241	0.80
	Symbolization	English	300	240	0.80
	Countermodel	–	300	44	0.15
qwen3-32B	Validity	Carroll	300	284	0.95
	Validity	English	300	292	0.97
	Symbolization	Carroll	300	247	0.82
	Symbolization	English	300	255	0.85
	Countermodel	–	300	261	0.89
anthropic_claude-3.7-sonnet	Validity	Carroll	300	287	0.96
	Validity	English	300	282	0.94
	Symbolization	Carroll	300	214	0.71
	Symbolization	English	300	228	0.76
	Countermodel	–	300	126	0.42
openai/gpt-4o	Validity	Carroll	300	262	0.87
	Validity	English	300	256	0.85
	Symbolization	Carroll	300	223	0.74
	Symbolization	English	300	217	0.72
	Countermodel	–	300	30	0.10
google_gemini-2.5-flash	Validity	Carroll	300	288	0.96
	Validity	English	300	283	0.94
	Symbolization	Carroll	300	229	0.76
	Symbolization	English	300	238	0.79
	Countermodel	–	300	40	0.13
deepseek_chat	Validity	Carroll	300	244	0.81
	Validity	English	300	229	0.76
	Symbolization	Carroll	300	240	0.80
	Symbolization	English	300	242	0.81
	Countermodel	–	300	104	0.35
microsoft_phi-4	Validity	Carroll	300	239	0.80
	Validity	English	300	220	0.73
	Symbolization	Carroll	300	152	0.51
	Symbolization	English	300	172	0.57
	Countermodel	–	300	43	0.14

Table 3: Accuracy across tasks, languages, and models.