

# LARGE LANGUAGE MODELS MEET SYMBOLIC PROVERS FOR LOGICAL REASONING EVALUATION

**Anonymous authors**

Paper under double-blind review

## ABSTRACT

First-order logic (FOL) reasoning, which involves sequential deduction, is pivotal for intelligent systems and serves as a valuable task for evaluating reasoning capabilities, particularly in chain-of-thought (CoT) contexts. Existing benchmarks often rely on extensive human annotation or handcrafted templates, making it difficult to achieve the necessary complexity, scalability, and diversity for robust evaluation. To address these limitations, we propose a novel framework that synergizes the generative strengths of Large Language Models (LLMs) with the rigor and precision of symbolic provers, enabling the creation of a scalable, diverse, and high-quality FOL reasoning dataset, ProverGen. ProverGen is also distinguished by its inclusion of accessible and logically coherent intermediate reasoning steps for each problem. Our evaluation shows that state-of-the-art LLMs struggle to solve ProverGen problems, even with CoT prompting, highlighting the dataset’s challenging nature. We also finetune Llama3.1-8B-Instruct on a separate training set generated by our framework. The finetuned model demonstrates consistent improvements on both in-distribution and out-of-distribution test sets, suggesting the value of our proposed data generation framework.<sup>1</sup>

## 1 INTRODUCTION

First-order logic (FOL) reasoning, which involves sequential deduction and the application of facts and logical rules to derive conclusions or make decisions, is a fundamental aspect of human intelligence (Saparov & He, 2022; Pan et al., 2023). The complexity of performing multi-step reasoning while maintaining logical coherence across these steps makes FOL reasoning an ideal testbed for evaluating the reasoning capabilities of Large Language Models (LLMs), particularly in scenarios such as Chain-of-Thought (CoT) prompting (Wei et al., 2022) and planning (Wang et al., 2024; Valmeekam et al., 2022). An effective benchmark for FOL reasoning should satisfy several key criteria: (1) *Scalability*, enabling expansion in both volume and complexity with minimal manual intervention; (2) *Natural and Diverse Language*, capturing a wide range of natural language expressions to reflect real-world linguistic variability; (3) *Symbolic Representations*, providing formal symbolic structures that can be validated through automated symbolic provers, ensuring dataset integrity and supporting downstream tasks like NL-FOL translation and tool based logic problem solving (Pan et al., 2023; Olausson et al., 2023); and (4) *Faithful Reasoning Chains*, where each instance includes intermediate reasoning steps clearly articulated in both symbolic and natural language formats, promoting transparency and facilitating further model training.

However, existing FOL reasoning datasets only partially fulfill these requirements (see Table 1 for more details). Template-based datasets such as ProntoQA (Saparov & He, 2022; Saparov et al., 2024), ProofWriter (Tafjord et al., 2021), and RuleTaker (Clark et al., 2021) are scalable but often lack diversity and rely on overly simplistic rules. On the contrary, manually curated datasets like FOLIO (Han et al., 2022) offer rich diversity in natural language expressions and more complex logic rules, but they are limited in size due to the extensive human effort needed for creation. [Revision: Additionally, data contamination is also a significant issue in manually annotated datasets, as it is difficult to update them frequently. This limitation can lead to biased evaluations and hinder true generalization due to potential data leakage.] Another common limitation across previous datasets is the lack of well-defined and easily accessible reasoning chains. FOLIO, for instance, does not

<sup>1</sup>We will release our code and dataset.

Table 1: Comparison of existing FOL reasoning datasets with ProverGen.

| Dataset      | Creation Method | Scalability | Natural & Diverse Language | Symbolic Representations | Faithful Reasoning Chains |
|--------------|-----------------|-------------|----------------------------|--------------------------|---------------------------|
| RuleTaker    | Synthetic       | ✓           | ✗                          | ✗                        | ✓                         |
| ProofWriter  | Synthetic       | ✓           | ✗                          | ✗                        | ✓                         |
| ProntoQA     | Synthetic       | ✓           | ✗                          | [Revision: ✓]            | ✓                         |
| ProntoQA-OOD | Synthetic       | ✓           | ✗                          | [Revision: ✓]            | ✓                         |
| LogicNLI     | Synthetic       | ✓           | ✗                          | ✗                        | ✗                         |
| FOLIO        | Manual          | ✗           | ✓                          | ✓                        | ✗                         |
| ProverGen    | Synthetic       | ✓           | ✓                          | ✓                        | ✓                         |

provide any reasoning chains, while ProofWriter and RuleTaker include reasoning chains, but they are not readily accessible and require users to implement additional code to extract them for each problem.

To address these gaps, we propose a novel framework that leverages the strengths of LLMs and symbolic provers to generate an extensible, diverse, and high-quality dataset for FOL reasoning. While LLMs have been increasingly utilized for synthesizing datasets across various tasks (Taori et al., 2023; Acharya et al., 2023; Gajjar & Shah, 2024), they fall short in producing reliable reasoning, often lacking faithfulness and transparency (Golovneva et al., 2022; Ribeiro et al., 2022; Lyu et al., 2023), particularly when it comes to long reasoning chains. On the other hand, symbolic provers are known for their reliability and precision in executing complex logical inferences. By combining the generative capabilities of LLMs with the rigor of symbolic provers, our framework offers a robust solution for producing FOL reasoning data that ensures both diversity and logical soundness. [Revision: It also mitigates the data contamination challenge by enabling the generation of new datasets using diverse models and controlled complexity, ensuring that the datasets remain fresh and uncontaminated.] Our framework follows a structured three-step pipeline: (1) we first use an LLM to generate a unique background story for each problem based on a given subject and a characteristic keyword; (2) we employ a symbolic prover (i.e., Prover9) to construct reasoning trees and compute the truth values of relevant facts; and (3) we utilize an LLM to translate these logical expressions into natural language statements.

We use Llama3.1-70B-Instruct within our proposed generation framework to create a new FOL reasoning benchmark, **ProverGen**, consisting of 1,500 instances across three difficulty levels: *easy*, *medium*, and *hard*. We evaluate ProverGen on several state-of-the-art (SOTA) LLMs and find that even with CoT prompting, models barely exceed 50% accuracy on the hard subset, highlighting the significant challenge posed by ProverGen. [Revision: ProverGen’s difficulty settings are carefully designed to ensure appropriate complexity], avoiding both performance saturation and uninformative under-performance (e.g., random guessing), enabling accurate evaluation of both powerful and smaller language models’ reasoning abilities. To further explore the potential of our proposed generation framework, we finetune Llama3.1-8B-Instruct on a separately generated ProverGen training set and observe substantial performance gains on in-distribution (ID) evaluations. Remarkably, this finetuned model also shows consistent improvements on out-of-distribution (OOD) test sets, significantly outperforming the same model finetuned on other existing FOL training sets.

## 2 RELATED WORKS

**Logical Reasoning Datasets** Logical reasoning is a fundamental ability of intelligent systems and researchers have proposed a number of datasets to evaluate logical reasoning abilities of language models. These datasets can be divided into two groups: manually crafted ones and automatically generated ones. Human-crafted datasets are typically sourced from existing examinations (Yu et al., 2019; Liu et al., 2021; 2023; Zhong et al., 2021) or directly written by domain experts (Hu et al., 2023; Han et al., 2022). Although they often exhibit higher quality and complexity, they also suffer from limited quantity and high creation costs. Automatically generated datasets are often created using rules and predefined natural language templates (Sinha et al., 2019; Hahn et al., 2021; Clark

et al., 2021; Tafjord et al., 2021; Tian et al., 2021; Saparov & He, 2022; Saparov et al., 2024; Sun et al., 2021). While these datasets enable automatic problem generation, they often lack linguistic diversity and are constrained by rigid, monotonous rules. None of them cover all aspects of first-order logic, and they cannot be used to generate problems with more complex rules.

**Symbolic Prover Augmented LLMs** Symbolic provers are computer programs used to generate proofs and solutions of mathematical theorems automatically. In logical reasoning problems, symbolic provers, such as SymPy (Meurer et al., 2017), Z3 (De Moura & Bjørner, 2008), Lean (De Moura et al., 2015), Pyke (Frederiksen, 2008) and Prover9 (McCune, 2005), stand out with faithful solutions and traceable intermediate processes. Recent works (Pan et al., 2023; Olausson et al., 2023; Ye et al., 2024; Zhang et al., 2024) combines LLMs and symbolic provers to solve logic reasoning problems. They use LLMs to translate problems from natural language into symbolic representations and then solve that problem with a symbolic prover. In contrast to previous works that utilize symbolic provers solely as problem solvers, our approach integrates symbolic provers into the generation process of first-order logic (FOL) problems. This integration allows us to bypass the intricate interrelations and couplings among logical relations, enabling the generation of diverse rules and their correct chaining to form comprehensive problems.

### 3 METHODOLOGY

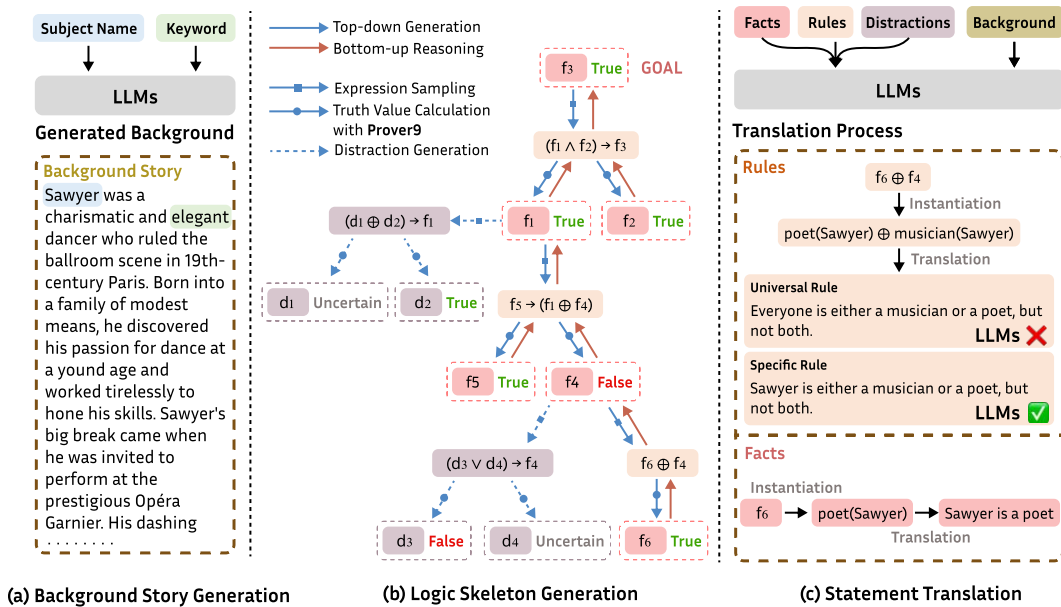


Figure 1: An overview of our LLMs-ProverGen framework. (a) **Background Story Generation** (Section 3.2.1). Given a subject, *Sawyer*, and a seed keyword, *elegant*, LLMs generate a background story to establish context and ensure linguistic diversity. (b) **Logic Skeleton Generation** (Section 3.2.2). A top-down approach is used to generate the logic skeleton, forming the reasoning tree for the FOL problem. This involves two iterative steps: expression sampling and truth value calculation using the Prover9 prover. Distractions are also incorporated to test the robustness of model’s reasoning capabilities. (c) **Statement Translation** (Section 3.2.3). LLMs translate the facts and rules from the logic skeleton into natural language, guided by the previously generated background story. Each rule is translated to a universal version and a specific version. The universal version is preferred if it does not contradict with common sense, otherwise the specific one is selected.

### 3.1 TASK FORMULATION

Given a set of premises  $\mathcal{P}$ , consisting of facts  $\mathcal{F}$  and rules  $\mathcal{R}$ , denoted as  $\mathcal{P} = (\mathcal{F}, \mathcal{R})$ , the goal of FOL reasoning is to determine whether a given goal  $\mathcal{G}$  (which can be a fact or a rule) is True, False, or Uncertain. Specifically,  $\mathcal{F} = \{f_1, f_2, \dots, f_m\}$  represents a set of facts, where each fact  $f_i$  ( $1 \leq i \leq m$ ) is a declarative statement describing attributes or characteristics of subjects. For example, Sawyer has good dance skills is a fact that specifies the attribute (has good dance skills) of the subject (Sawyer). Similarly,  $\mathcal{R} = \{r_1, r_2, \dots, r_n\}$  comprises a set of rules, where each rule  $r_j$  ( $1 \leq j \leq n$ ) defines relationships between multiple facts using seven FOL symbols: conjunction ( $\wedge$ ), disjunction ( $\vee$ ), negation ( $\neg$ ), implication ( $\rightarrow$ ), equivalence ( $\equiv$ )<sup>2</sup>, universal quantifier ( $\forall$ ), and existential quantifier ( $\exists$ ). For example, the rule If Sawyer has good dance skills and has charisma, then he is a successful dancer illustrates a relationship between two facts and their conclusion. It is worth noting that some existing datasets, such as ProntoQA (Saparov & He, 2022) and ProofWriter (Tafjord et al., 2021), do not fully encompass all seven FOL relationships, whereas our proposed framework ensures complete coverage.

### 3.2 LLM & PROVER-EMPOWERED GENERATION FRAMEWORK

Our proposed generation framework, which integrates the generative capabilities of LLMs with the precision of a symbolic prover, generates FOL reasoning problems through a structured three-stage process, as illustrated in Figure 1. First, a background story is created for each problem using sampled subject names and keywords to ensure linguistic diversity. In the next stage, the logic skeleton generation employs a symbolic prover (i.e., Prover9) to construct reasoning trees and determine the truth values of the relevant facts. Finally, the statement translation phase uses LLMs to convert these logical expressions, including both premises and goals, into natural language, ensuring clarity and coherence. An example of a generated FOL problem is shown in Figure 2.

#### 3.2.1 BACKGROUND STORY GENERATION

Achieving naturalness in language expression is crucial for creating a plausible FOL dataset. Existing approaches often rely on extensive human annotation or rigidly structured templates. For instance, ProntoQA (Saparov & He, 2022) and ProofWriter (Tafjord et al., 2021) employ handcrafted templates, which result in datasets lacking linguistic diversity and naturalness.<sup>3</sup>

In contrast, FOLIO (Han et al., 2022) retrieves documents from Wikipedia and manually extracts facts and rules. However, this process requires substantial human effort and is susceptible to syntactic typos and reasoning errors introduced during annotation, as observed in FOLIO. Additionally, models may exploit *shortcuts* by leveraging pre-trained knowledge to predict answers directly, bypassing rigorous logical reasoning (Qi et al., 2023; Geirhos et al., 2020), since the facts and rules in the dataset are often derived from the real-life.

To efficiently synthesize plausible natural FOL problems, we leverage LLMs to generate a unique background story for the subject of each problem, which serves to instantiate the statements (both premises and goals) within the problem. However, simply prompting LLMs to generate stories without any guidance can be suboptimal, as they struggle to produce diversified content with high quality, as noted by Jentzsch & Kersting (2023); Sudalairaj et al. (2024). To mitigate these issues, we prepare a characteristic keyword for the subject within the problem to guide the generation. Specifically, the process begins by assigning a unique name to the subject, which is randomly sampled from publicly accessible datasets of human names<sup>4</sup> and pet names<sup>5</sup>. Next, we sample a keyword

<sup>2</sup>In line with previous datasets, we replace equivalence ( $\equiv$ ) with exclusive disjunction ( $\oplus$ ), which can be interpreted as a combination of equivalence ( $\equiv$ ) and negation ( $\neg$ ):  $f_1 \oplus f_2 = \neg(f_1 \equiv f_2)$

<sup>3</sup>Both ProofWriter and PrOntoQA generate natural language expressions by using predefined templates for each logical rule. For example, for a rule like  $A \rightarrow B$ , they may define a template such as Every A is B, and then randomly select words to replace A and B. [Revision: PrOntoQA intentionally avoids real-world concepts to prevent conflicts with real-world knowledge. They also evaluate LLMs' behavior on examples that contain facts and rules that are consistent (or inconsistent) with the real-world knowledge. In contrast, ProofWriter does not incorporate such a mechanism.]

<sup>4</sup><https://github.com/aruljohn/popular-baby-names>

<sup>5</sup><https://github.com/fregante/pet-names/blob/master/pet-names.json>

from WordNet (Miller, 1995) to serve as a defining characteristic or attribute of the subject, ensuring semantic diversity in the generated content. These elements, the name and the characteristic keyword, are then integrated into the prompt to guide LLMs to generate a contextually rich and varied background story. This controlled generation helps promote diversity in the dataset. The detailed prompt used in our experiments is outlined in Appendix A.

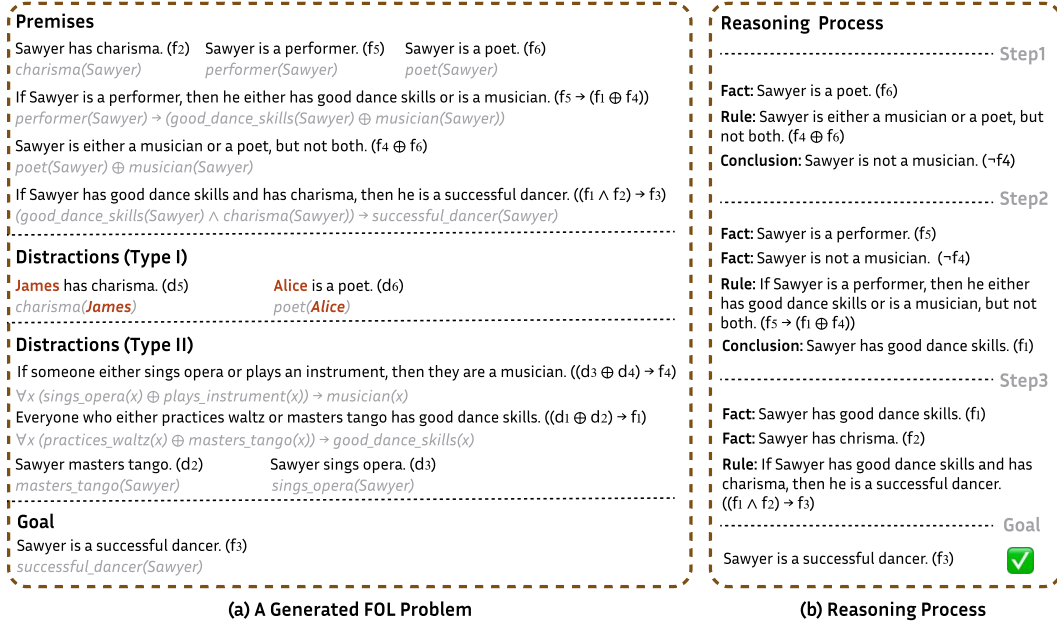


Figure 2: A generated FOL problem (a) with its corresponding reasoning process (b).

### 3.2.2 LOGIC SKELETON GENERATION

Logical soundness is a fundamental requirement in constructing a robust FOL reasoning dataset. While LLMs are highly proficient at generating natural language, they often fall short in maintaining logical coherence, particularly when dealing with intricate, multi-step reasoning processes. To address this limitation, we introduce a novel approach that integrates symbolic prover into the data generation pipeline, allowing for the construction of logically rigorous FOL reasoning problems.

In our data generation framework, we start with constructing a logic skeleton for each problem, represented as a reasoning tree with the ultimate goal at its root (as shown in Figure 1(b)). Previous works typically generate reasoning problems in a *bottom-up* approach by starting with premises and building toward conclusions. [Revision: We argue that this approach could be flawed as it can often result in disconnected or contradictory premises, making it difficult to ensure that all facts are logically coherent with each other. For instance, if the problem includes facts such as  $f_1$ : Sawyer is a poet,  $f_2$ : Sawyer comes from Mars, and  $f_3$ : Sawyer is a large, fluffy Maine Coon cat, then integrating these elements into a consistent reasoning chain is challenging.] Therefore, our framework adopts a novel *top-down* approach, which begins with the goal and works backward to derive the relevant premises and rules needed to logically support that goal. This method ensures that each fact and rule contributes directly to the reasoning process, improving both the coherence and logical soundness of the generated problems. Moreover, this top-down structure simplifies the problem of logical consistency by allowing LLMs to focus on facts that are directly relevant to the conclusion, without needing to reconcile them with unrelated information.

More specifically, as shown in Figure 1(b), we begin by setting the truth value of the goal, denoted as  $f_3 = \text{True}$ , and treat this as the root node of the reasoning tree. Next, we sample a logical expression for this node,  $(f_1 \wedge f_2) \rightarrow f_3$ , which establishes a relationship between the goal  $f_3$  and the premises  $f_1$  and  $f_2$ . At this stage, we use the symbolic prover Prover9 to calculate the truth values of  $f_1$  and  $f_2$ , ensuring that the premises logically support the goal. Once  $f_1$  and  $f_2$  are assigned truth values, they become the current nodes in the reasoning tree. For each of these nodes, we continue

the process by sampling new logical expressions. For instance, for  $f_1 = \text{True}$ , we sample a new rule  $f_5 \rightarrow (f_1 \oplus f_4)$ , where  $f_5$  and  $f_4$  are additional facts to be evaluated. We then calculate the truth values for these new premises using Prover9, and they become the next current nodes in the tree. This iterative process is repeated for each current node, expanding the reasoning tree step by step. The process continues until we have generated the desired number of reasoning steps, ensuring that each layer of the reasoning tree is logically consistent and fully developed, forming a coherent and complete logical structure for the problem.

To further assess the robustness of models in handling FOL reasoning, we introduce distractions into the generated problems. Distractions, denoted as  $d$ , are an additional set of facts and rules not essential to solving the problem. The inclusion of distractions forces models to distinguish between relevant and distracting information. We implement two types of distractions in our framework. The first type follows existing works (Tafjord et al., 2021; Saparov & He, 2022), where we randomly introduce facts or rules that are related to the problem’s domain but use a different subject name (see Figure 2). The second, a more complex type of distraction leverages the power of symbolic provers to introduce facts and rules that are semantically or logically related to the reasoning chain, but which lead to an Uncertain judgment. This type of distraction requires models to reason over logically relevant information that does not directly impact the truth value of the core facts needed to reach the final conclusion. For instance, suppose  $f_1$  is a critical fact in the reasoning tree, and we introduce two distractions,  $d_1 = \text{True}$  and  $d_2 = \text{Uncertain}$ , with the rule  $(d_1 \oplus d_2) \rightarrow f_1$ . Here, the truth value of  $f_1$  remains Uncertain because the rule involves incomplete information. Models must reason through these distractions, yet ultimately recognize that they do not alter the critical facts that drive the reasoning chain forward.

### 3.2.3 STATEMENT TRANSLATION

After generating the logical skeleton of a problem, we use LLMs to translate each rule and fact into natural language within the framework of the background story generated in Section 3.2.1. As illustrated in Figure 1(c), we first prompt LLMs to instantiate placeholders (such as  $f_6$  and  $f_4$ ) in the logic expressions with appropriate predicates, like `poet` and `musician`. Once the placeholders are instantiated, the logic expressions are translated into natural language. To maintain consistency throughout the translation process, we provide the previously translated facts and rules as references in the prompt for subsequent translations (details of the translation prompt can be found in Appendix B).

To ensure the plausibility and soundness of the rules, we instruct LLMs to generate two versions of each rule: a universal and a specific version. The universal rule refers to statements that apply to everyone or everything, such as `Everyone is either a musician or a poet, but not both`. The specific rule, on the other hand, applies to an individual subject, such as `Sawyer is either a musician or a poet, but not both`. LLMs then assess whether the universal rule aligns with common sense. If it does, we use the universal version; if not, [Revision: we opt for the specific rule]. For instance, the universal rule in Figure 1(c) does not align with common sense (as there are more professions than just musicians and poets), so we choose the specific version of the rule. The final stage of the proposed framework involves assembling the facts, rules, and distractions generated in previous stages into a complete FOL problem. An example of the constructed problem with its corresponding reasoning steps is provided in Figure 2.

## 3.3 THE PROVERGEN BENCHMARK

We utilize Llama3.1-70B-Instruct throughout our proposed generation framework to develop a new FOL reasoning benchmark, **ProverGen**, comprising 1,500 instances. The benchmark is evenly divided into three subsets based on the length of the reasoning chains: *easy* (1-2 steps), *medium* (3-5 steps), and *hard* (6-9 steps). In addition to the number of reasoning steps, the *hard* subset is characterized by more diverse reasoning directions. Reasoning in the *easy* and *medium* subsets typically follows a straightforward left-to-right progression. For example, given  $f_1 = \text{True}$  and  $f_1 \rightarrow f_2$ , the model can deduce that  $f_2 = \text{True}$ . In contrast, the *hard* subset involves more complex patterns, requiring models to infer  $f_1 = \text{False}$  given  $f_2 = \text{False}$  and  $f_1 \rightarrow f_2$ , thus demonstrating a more intricate reasoning process. Each problem in the benchmark contains two types of distractions (see Section 3.2.2), where the number of distractions for each type is randomly sampled. Moreover, to further challenge the robustness of models, we shuffle the order of the premises within each

problem, ensuring that models cannot exploit fixed patterns or sequences to derive conclusions. Detailed examples of the ProverGen are provided in Appendix D.

## 4 EVALUATION ON THE PROVERGEN

### 4.1 EXPERIMENTAL SETUP

We evaluate current SOTA proprietary LLMs (GPT-4o (OpenAI, 2024) and Claude3.5 (Anthropic, 2024))<sup>6</sup> and open-source LLMs (Mistral-7B-Instruct, Mistral-Large-Instruct, Mixtral-8x22B-Instruct-v0.1 (Jiang et al., 2023; 2024) and Llama3.1-8B-Chat, Llama3.1-70B-Chat (Dubey et al., 2024)) on the ProverGen. We utilize two prompting strategies: *Standard* and *CoT*. *Standard* prompting uses 2-shot in-context learning to prompt LLMs to answer questions directly, whereas *CoT* prompting employs 2-shot examples to instruct the model to solve questions step by step. Examples of both prompting strategies can be found in Appendix E.

### 4.2 MAIN RESULTS

Table 2: The performance of LLMs on ProverGen and existing FOL reasoning datasets. †: The average performance improvement of *CoT* prompting over *Standard* prompting across models on each evaluation set, which indicates the average gain *CoT* prompting brings on each set.

|                           | ProverGen |        |       | ProntoQA | ProofWriter | FOLIO | Avg   |
|---------------------------|-----------|--------|-------|----------|-------------|-------|-------|
|                           | Easy      | Medium | Hard  |          |             |       |       |
| <i>Standard Prompting</i> |           |        |       |          |             |       |       |
| GPT-4o                    | 87.20     | 68.60  | 46.20 | 91.80    | 56.33       | 67.86 | 69.66 |
| Claude-3.5-Sonnet         | 85.00     | 68.20  | 42.80 | 88.60    | 55.00       | 77.85 | 69.58 |
| Llama3.1-8B-Instruct      | 46.60     | 43.00  | 39.00 | 50.40    | 43.80       | 54.29 | 46.18 |
| Llama3.1-70B-Instruct     | 82.00     | 64.20  | 47.60 | 80.60    | 50.33       | 67.86 | 65.28 |
| Mistral-7B-Instruct       | 56.80     | 46.80  | 37.20 | 50.00    | 42.33       | 54.29 | 47.90 |
| Mistral-Large-Instruct    | 84.60     | 69.20  | 49.60 | 71.00    | 60.33       | 77.14 | 68.65 |
| Mixtral-8x22B-Instruct    | 75.40     | 57.40  | 39.00 | 65.20    | 40.17       | 74.29 | 58.58 |
| <i>CoT Prompting</i>      |           |        |       |          |             |       |       |
| GPT-4o                    | 94.20     | 79.40  | 50.00 | 100.00   | 67.33       | 72.14 | 77.18 |
| Claude-3.5-Sonnet         | 95.20     | 83.60  | 56.40 | 99.20    | 76.33       | 80.71 | 81.91 |
| Llama3.1-8B-Instruct      | 75.60     | 46.60  | 33.60 | 79.60    | 56.83       | 63.57 | 59.30 |
| Llama3.1-70B-Instruct     | 90.40     | 73.20  | 46.80 | 95.40    | 71.17       | 74.29 | 75.21 |
| Mistral-7B-Instruct       | 72.00     | 51.00  | 41.80 | 61.20    | 46.00       | 63.58 | 55.93 |
| Mistral-Large-Instruct    | 92.60     | 75.80  | 52.20 | 98.60    | 73.50       | 83.57 | 79.38 |
| Mixtral-8x22B-Instruct    | 87.60     | 66.80  | 47.60 | 79.60    | 57.67       | 73.57 | 68.81 |
| Avg $\Delta^\dagger$      | 12.86     | 8.43   | 3.86  | 16.57    | 14.36       | 5.41  | 10.25 |

**ProverGen as a New Challenge** Overall, model performance on ProverGen declines as problem difficulty increases, with the lowest scores observed on the hard subset *across all existing datasets* as seen in Table 2. Specifically, strong models such as GPT-4o, Claude-3.5-Sonnet and Mistral-Large-Instruct barely exceed 50% accuracy on the hard subset, even with CoT prompting. This result indicates that even the most advanced LLMs perform poorly on sufficiently complex problems with long reasoning chains. In contrast, simpler benchmarks like ProntoQA appear to be nearly solved by models like GPT-4o and Claude-3.5-Sonnet. However, smaller models still struggle to achieve satisfactory performance, even on the easy subset of ProverGen, suggesting that these models are not yet equipped to handle FOL reasoning effectively.

<sup>6</sup>The version of GPT-4o we used in our experiments is gpt-4o-2024-08-06, and the version of Claude3.5 we used is claude-3-5-sonnet-20240620.

**Comparison of prompting strategies** Compared with *Standard* prompting, *CoT* prompting notably enhances the performance of models on easy datasets but does not yield a similar improvement on harder datasets, as presented in the last row of Table 2. The improvement on the easy dataset is more pronounced because performance on easy problems is probably limited by the CoT capability, whereas the bottleneck for hard problems may involve other factors beyond CoT capability. [Revision: We have also conducted experiments using 5-shot examples; the results can be found in Appendix G.]

#### 4.3 ABLATION STUDY ON DISTRACTIONS AND PREMISE ORDER

By default, the ProverGen benchmark includes two types of distractions and shuffles the order of premises within each problem (see Section 3.3). To assess how these two factors influence the evaluation results of LLMs, we conduct ablation studies using GPT-4o with *CoT* prompting in this section. Specifically, in the first setting, we remove all distractions from the problems to observe how the absence of irrelevant information affects the model’s reasoning performance. Results in Table 3 exhibit that the model performs better in this situation. In the second setting, we *additionally* reorder the premises to match the logical progression of the underlying reasoning chain. This turns out that the model performs even better when premises are presented in a coherent sequence. Consequently, these findings suggest both factors are valuable for constructing a robust FOL reasoning benchmark for LLMs.

Table 3: Performance of GPT-4o on different variants of ProverGen using CoT prompting. The first row represents the default ProverGen setting, which features distractions and shuffled premises.

| <b>Distractions</b> | <b>Shuffled Premises</b> | <b>Easy</b> | <b>Medium</b> | <b>Hard</b> |
|---------------------|--------------------------|-------------|---------------|-------------|
| ✓                   | ✓                        | 94.20       | 79.40         | 50.00       |
| ✗                   | ✓                        | 94.40       | 82.20         | 57.40       |
| ✗                   | ✗                        | 95.60       | 87.40         | 62.40       |

## 5 FINETUNING WITH PROVERGEN TRAIN DATA

Beyond its role as an evaluation benchmark, we also investigate whether the training data synthesized by our proposed framework can improve a model’s FOL reasoning capabilities, particularly in out-of-distribution (OOD) scenarios. To this end, we generate a dedicated training set and finetune an open-source model, Llama3.1-8B-Instruct. For comparison, we also finetune the same model on existing FOL reasoning datasets, such as ProofWriter and FOLIO. We exclude ProntoQA from this experiment due to the absence of a publicly available training set. Detailed training configurations are provided in Appendix F.

### 5.1 EVALUATION RESULTS

**In-Distribution (ID) Performance** As shown in Table 4, finetuning on the ProverGen training set results in a substantial performance improvement across all subsets of the ProverGen benchmark. Specifically, the model’s performance increases by 21.4%, 44.0%, and 34.6% in the easy, medium, and hard subsets of the ProverGen benchmark, respectively. Similarly, fine-tuning on ProofWriter and FOLIO leads to performance gains on their corresponding test sets, as expected.

**Out-of-Distribution (OOD) Performance** While the ID performance of finetuned models can exhibit the positive effects of specialized finetuning, OOD performance provides more insights as it demonstrates the generalizability of the finetuned models. As shown in Table 4, finetuning on ProverGen yields a performance increase on OOD test sets (ProntoQA, ProofWriter, FOLIO) by about 5% to 8.8%. Finetuning on ProverGen obtains the best OOD improvement on average with a large leading margin compared to models finetuned on ProofWriter and FOLIO, indicating the limited utility of these prior datasets for promoting the reasoning capabilities of LLMs. The model finetuned on ProofWriter achieves the highest performance on ProntoQA. This anomaly could be



Table 4: Performance of Llama3.1-8B-Instruct finetuned with different FOL training sets. Green cells indicate in-distribution (ID) evaluation results, while red cells represent out-of-distribution (OOD) evaluation results. The change of performance after finetuning is shown in small fonts along with the evaluation results. †: the average *change* of performance across all OOD evaluation sets for each training configuration.

| FOL Training Set                      | ProverGen                     |                               |                               | ProntoQA                      | ProofWriter                   | FOLIO                        | OOD Avg $\Delta^\dagger$ |
|---------------------------------------|-------------------------------|-------------------------------|-------------------------------|-------------------------------|-------------------------------|------------------------------|--------------------------|
|                                       | Easy                          | Medium                        | Hard                          |                               |                               |                              |                          |
| <i>Llama3.1-8B-Instruct</i>           |                               |                               |                               |                               |                               |                              |                          |
| -                                     | 75.60                         | 46.60                         | 33.60                         | 79.60                         | 56.83                         | 63.57                        | -                        |
| <i>Finetuned Llama3.1-8B-Instruct</i> |                               |                               |                               |                               |                               |                              |                          |
| ProofWriter                           | 44.60 <sup>-31.0</sup>        | 55.00 <sup>+8.4</sup>         | 47.20 <sup>+13.6</sup>        | <b>92.00</b> <sup>+12.4</sup> | <b>71.67</b> <sup>+14.8</sup> | 50.00 <sup>-13.6</sup>       | -2.0                     |
| FOLIO                                 | 53.20 <sup>-22.4</sup>        | 44.80 <sup>-1.8</sup>         | 31.00 <sup>-2.6</sup>         | 63.40 <sup>-16.2</sup>        | 42.83 <sup>-14.0</sup>        | <b>70.00</b> <sup>+6.4</sup> | -11.4                    |
| ProverGen                             | <b>97.00</b> <sup>+21.4</sup> | <b>90.60</b> <sup>+44.0</sup> | <b>68.20</b> <sup>+34.6</sup> | 88.40 <sup>+8.8</sup>         | 65.67 <sup>+8.8</sup>         | 68.57 <sup>+5.0</sup>        | <b>+7.5</b>              |

attributed to the high similarity between ProntoQA and ProofWriter: Both datasets utilize predefined templates for generating logic and corresponding natural language.

## 5.2 CASE STUDY

We show examples from the FOLIO test set in Table 5 to demonstrate how finetuning on ProverGen improves model performance. Before finetuning, models fail to solve the problems because of various reasons, such as using a common sense shortcut, being unfaithful to the context, and being unable to extend the reasoning chain. After training on ProverGen, models are able to derive a valid reasoning chain up to the goal based on the context, and the reasoning process rigidly adheres to the facts and rules.

## 6 CONCLUSIONS

In this work, we presented a novel framework for automatically generating FOL reasoning datasets by leveraging the strengths of LLMs in producing diverse natural language expressions and the rigor of symbolic provers in ensuring logical consistency. Using this framework, we developed ProverGen, a high-quality FOL reasoning benchmark that is both extensible and diverse. ProverGen encompasses complex rules, diverse natural language expressions, and faithful intermediate reasoning chains, providing a comprehensive evaluation benchmark for FOL reasoning. Our evaluation shows that SOTA LLMs struggle to solve ProverGen problems, even with CoT prompting, demonstrating the unique challenge posed by our dataset. We also finetuned Llama3.1-8b-Instruct on a separately generated training set and observed consistent performance improvements on both in-distribution and out-of-distribution test sets. We hope that our generation framework and the resulting dataset could contribute to the development of more robust and capable models in this area.

## 7 ETHICS STATEMENT

The keywords and names used in this paper are sourced entirely from public repositories and databases. Specifically, the names were collected from two public GitHub repositories under MIT licenses, and the keywords were obtained from WordNet. All sources have licenses permitting free usage, and our work complies with these terms. Our dataset does not involve human participants, nor does it present any immediately harmful insights. For the release, we plan to open source the code for creating the benchmark and the dataset we created for the experiments in this paper.

Table 5: Typical errors made by models and how finetuning correct them. Reasoning chains before and after finetuning are both displayed for comparison.

| Error Type                           | Context  | Before Finetuning   | After Finetuning   |
|--------------------------------------|--|---|--|
| Common sense shortcut                | All students are humans. If someone studies, then he/she is a student. A person either studies or teaches. No young person teaches. Jerry neither teaches nor is a manager. Rose is young or a student. Question: Jerry is not young.  | A person either studies or teaches. No young person teaches. Therefore, a young person studies. If someone studies, then they are a student. All students are humans. Therefore, a young person is a human. People are either young or old. Therefore, a young person is not old. Jerry neither teaches nor is a manager. Therefore, Jerry studies. Since Jerry is a student and a human, Jerry is not old. Therefore, Jerry is young. Answer: false.             | Jerry neither teaches nor is a manager. A person either studies or teaches. Jerry studies. Jerry studies. If someone studies, then he/she is a student. Jerry is a student. Jerry is a student. All students are humans. Jerry is a human. According to the context and the conclusions that have already been drawn, we can deduce that it is uncertain that Jerry is not young. Answer: uncertain. |
| Unfaithful to the context            | Beijing is the capital of the People's Republic of China. Beijing is the world's most populous national capital city. Beijing is located in Northern China. Beijing hosted the 2008 Summer Olympics and 2008 Summer Paralympics Games. Beijing has hosted both the Summer and Winter Olympics, along with the Summer and Winter Paralympics. Question: Beijing has hosted both the 2008 Summer Olympics and a winter olympics. | Beijing hosted the 2008 Summer Olympics and 2008 Summer Paralympics Games. However, there is no information about Beijing hosting a Winter Olympics. Therefore, the statement is uncertain because it is not explicitly stated that Beijing has hosted a Winter Olympics, but it is also not explicitly stated that it has not. Answer: uncertain.  | Beijing hosted the 2008 Summer Olympics and 2008 Summer Paralympics Games. Beijing has hosted both the Summer and Winter Olympics, along with the Summer and Winter Paralympics. Beijing has hosted both the 2008 Summer Olympics and a winter olympics. Therefore, it is true that Beijing has hosted both the 2008 Summer Olympics and a winter olympics. Answer: true.                            |
| Unable to extend the reasoning chain | Robert Lewandowski is a striker. Strikers are soccer players. Robert Lewandowski left Bayern Munchen. If a player left a team they no longer play for that team. Question: Robert Lewandowski plays for Bayern Munchen.  | Robert Lewandowski is a striker. Strikers are soccer players. Therefore, Robert Lewandowski is a soccer player. Robert Lewandowski left Bayern Munchen. If a player left a team, they no longer play for that team. Therefore, Robert Lewandowski no longer plays for Bayern Munchen. However, the question is asking if he plays for Bayern Munchen, not if he left. The information does not state that he currently plays for another team. Answer: uncertain. | Robert Lewandowski left Bayern Munchen. If a player left a team they no longer play for that team. Robert Lewandowski does not play for Bayern Munchen. Therefore, it is false that Robert Lewandowski plays for Bayern Munchen. Answer: false.  |

## 8 REPRODUCIBILITY STATEMENT

For our submission, we have uploaded the entirety of the source code as a zipped file that has been properly anonymized. The source code contains inline documentation that details purpose and usage of different parts of the codebase. In addition, we also include the full set of model's responses and the evaluation script. As discussed in the ethics statement, we plan to more formally release ProverGen to the public as an open source repository with thorough details that describes the framewrok, outlines the code, and details its usage.

## REFERENCES

- Arkadeep Acharya, Brijraj Singh, and Naoyuki Onoe. Llm based generation of item-description for recommendation system. In *Proceedings of the 17th ACM Conference on Recommender Systems*, pp. 1204–1207, 2023.
- Anthropic. Claude 3.5 sonnet. 2024. URL <https://www.anthropic.com/news/claude-3-5-sonnet>.
- Peter Clark, Oyvind Tafjord, and Kyle Richardson. Transformers as soft reasoners over language. In *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, pp. 3882–3890, 2021.
- Leonardo De Moura and Nikolaj Bjørner. Z3: An efficient smt solver. In *International conference on Tools and Algorithms for the Construction and Analysis of Systems*, pp. 337–340. Springer, 2008.
- Leonardo De Moura, Soonho Kong, Jeremy Avigad, Floris Van Doorn, and Jakob von Raumer. The lean theorem prover (system description). In *Automated Deduction-CADE-25: 25th International Conference on Automated Deduction, Berlin, Germany, August 1-7, 2015, Proceedings 25*, pp. 378–388. Springer, 2015.

- 540 Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha  
541 Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models.  
542 *arXiv preprint arXiv:2407.21783*, 2024.
- 543
- 544 Bruce Frederiksen. Applying expert system technology to code reuse with pyke. *PyCon: Chicago*,  
545 2008.
- 546
- 547 Pranshav Gajjar and Vijay K Shah. Oran-bench-13k: An open source benchmark for assessing llms  
548 in open radio access networks. *arXiv preprint arXiv:2407.06245*, 2024.
- 549
- 550 Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel,  
551 Matthias Bethge, and Felix A Wichmann. Shortcut learning in deep neural networks. *Nature*  
552 *Machine Intelligence*, 2(11):665–673, 2020.
- 553
- 554 Olga Golovneva, Moya Peng Chen, Spencer Poff, Martin Corredor, Luke Zettlemoyer, Maryam  
555 Fazel-Zarandi, and Asli Celikyilmaz. Roscoe: A suite of metrics for scoring step-by-step reason-  
556 ing. In *The Eleventh International Conference on Learning Representations*, 2022.
- 557
- 558 Christopher Hahn, Frederik Schmitt, Jens U Kreber, Markus Norman Rabe, and Bernd Finkbeiner.  
559 Teaching temporal logics to neural networks. In *International Conference on Learning Representations*, 2021.
- 560
- 561 Simeng Han, Hailey Schoelkopf, Yilun Zhao, Zhenting Qi, Martin Riddell, Luke Benson, Lucy  
562 Sun, Ekaterina Zubova, Yujie Qiao, Matthew Burtell, et al. Folio: Natural language reasoning  
563 with first-order logic. *arXiv preprint arXiv:2209.00840*, 2022.
- 564
- 565 Shengding Hu, Yifan Luo, Huadong Wang, Xingyi Cheng, Zhiyuan Liu, and Maosong Sun. Won't  
566 get fooled again: Answering questions with false premises. *arXiv preprint arXiv:2307.02394*,  
567 2023.
- 568
- 569 Sophie Freya Jentsch and Kristian Kersting. Chatgpt is fun, but it is not funny! humor is still  
570 challenging large language models. In *61st Annual Meeting of the Association for Computational*  
571 *Linguistics, ACL 2023*, 2023.
- 572
- 573 Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot,  
574 Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al.  
575 Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.
- 576
- 577 Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bam-  
578 ford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, et al.  
579 Mixtral of experts. *arXiv preprint arXiv:2401.04088*, 2024.
- 580
- 581 Hanmeng Liu, Jian Liu, Leyang Cui, Zhiyang Teng, Nan Duan, Ming Zhou, and Yue Zhang. Logiqa  
582 2.0—an improved dataset for logical reasoning in natural language understanding. *IEEE/ACM*  
583 *Transactions on Audio, Speech, and Language Processing*, 31:2947–2962, 2023. doi: 10.1109/  
584 TASLP.2023.3293046.
- 585
- 586 Jian Liu, Leyang Cui, Hanmeng Liu, Dandan Huang, Yile Wang, and Yue Zhang. Logiqa: a chal-  
587 lenge dataset for machine reading comprehension with logical reasoning. In *Proceedings of the*  
588 *Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelli-*  
589 *gence*, pp. 3622–3628, 2021.
- 590
- 591 Qing Lyu, Shreya Havaldar, Adam Stein, Li Zhang, Delip Rao, Eric Wong, Marianna Apidianaki,  
592 and Chris Callison-Burch. Faithful chain-of-thought reasoning. In *The 13th International Joint*  
593 *Conference on Natural Language Processing and the 3rd Conference of the Asia-Pacific Chapter*  
*of the Association for Computational Linguistics (IJCNLP-AAACL 2023)*, 2023.
- 594
- 595 William McCune. Release of prover9. In *Mile high conference on quasigroups, loops and nonasso-*  
*ciative systems, Denver, Colorado*, 2005.

- 594 Aaron Meurer, Christopher P. Smith, Mateusz Paprocki, Ondřej Čertík, Sergey B. Kirpichev,  
595 Matthew Rocklin, Amit Kumar, Sergiu Ivanov, Jason K. Moore, Sartaj Singh, Thilina Rath-  
596 nayake, Sean Vig, Brian E. Granger, Richard P. Muller, Francesco Bonazzi, Harsh Gupta, Shivam  
597 Vats, Fredrik Johansson, Fabian Pedregosa, Matthew J. Curry, Andy R. Terrel, Štěpán Roučka,  
598 Ashutosh Saboo, Isuru Fernando, Sumith Kulal, Robert Cimrman, and Anthony Scopatz. Sympy:  
599 symbolic computing in python. *PeerJ Computer Science*, 3:e103, January 2017. ISSN 2376-5992.  
600 doi: 10.7717/peerj-cs.103. URL <https://doi.org/10.7717/peerj-cs.103>.
- 601 George A Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):  
602 39–41, 1995.
- 603
- 604 Theo X Olausson, Alex Gu, Ben Lipkin, Cedegao E Zhang, Armando Solar-Lezama, Joshua B  
605 Tenenbaum, and Roger P Levy. Linc: A neurosymbolic approach for logical reasoning by com-  
606 bining language models with first-order logic provers. In *The 2023 Conference on Empirical*  
607 *Methods in Natural Language Processing*, 2023.
- 608 OpenAI. Hello GPT-4o. 2024. URL <https://openai.com/index/hello-gpt-4o/>.
- 609
- 610 Liangming Pan, Alon Albalak, Xinyi Wang, and William Yang Wang. Logic-lm: Empower-  
611 ing large language models with symbolic solvers for faithful logical reasoning. *arXiv preprint*  
612 *arXiv:2305.12295*, 2023.
- 613
- 614 Chengwen Qi, Bowen Li, Binyuan Hui, Bailin Wang, Jinyang Li, Jinwang Wu, and Yuanjun Laili.  
615 An investigation of llms’ inefficacy in understanding converse relations. In *Proceedings of the*  
616 *2023 Conference on Empirical Methods in Natural Language Processing*, pp. 6932–6953, 2023.
- 617 Danilo Neves Ribeiro, Shen Wang, Xiaofei Ma, Henghui Zhu, Rui Dong, Deguang Kong, Juli-  
618 ette Burger, Anjelica Ramos, William Yang Wang, George Karypis, et al. Street: A multi-task  
619 structured reasoning and explanation benchmark. In *The Eleventh International Conference on*  
620 *Learning Representations*, 2022.
- 621
- 622 Abulhair Saparov and He He. Language models are greedy reasoners: A systematic formal analysis  
623 of chain-of-thought. In *The Eleventh International Conference on Learning Representations*,  
624 2022.
- 625 Abulhair Saparov, Richard Yuanzhe Pang, Vishakh Padmakumar, Nitish Joshi, Mehran Kazemi,  
626 Najoung Kim, and He He. Testing the general deductive reasoning capacity of large language  
627 models using ood examples. *Advances in Neural Information Processing Systems*, 36, 2024.
- 628
- 629 Koustuv Sinha, Shagun Sodhani, Jin Dong, Joelle Pineau, and William L Hamilton. Clutrr: A diag-  
630 nostic benchmark for inductive reasoning from text. In *Proceedings of the 2019 Conference on*  
631 *Empirical Methods in Natural Language Processing and the 9th International Joint Conference*  
632 *on Natural Language Processing (EMNLP-IJCNLP)*, pp. 4506–4515, 2019.
- 633 Shivchander Sudalairaj, Abhishek Bhandwadar, Aldo Pareja, Kai Xu, David D. Cox, and Akash  
634 Srivastava. Lab: Large-scale alignment for chatbots, 2024. URL <https://arxiv.org/abs/2403.01081>.
- 635
- 636 Haitian Sun, William W Cohen, and Ruslan Salakhutdinov. Conditionalqa: A complex reading  
637 comprehension dataset with conditional answers. *arXiv preprint arXiv:2110.06884*, 2021.
- 638
- 639 Oyvind Tafjord, Bhavana Dalvi, and Peter Clark. Proofwriter: Generating implications, proofs, and  
640 abductive statements over natural language. In *Findings of the Association for Computational*  
641 *Linguistics: ACL-IJCNLP 2021*, pp. 3621–3634, 2021.
- 642
- 643 Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy  
644 Liang, and Tatsunori B. Hashimoto. Stanford alpaca: An instruction-following llama model.  
645 [https://github.com/tatsu-lab/stanford\\_alpaca](https://github.com/tatsu-lab/stanford_alpaca), 2023.
- 646
- 647 Jidong Tian, Yitian Li, Wenqing Chen, Liqiang Xiao, Hao He, and Yaohui Jin. Diagnosing the  
648 first-order logical reasoning ability through logicnli. In *Proceedings of the 2021 Conference on*  
649 *Empirical Methods in Natural Language Processing*, pp. 3738–3747, 2021.

648 Karthik Valmeekam, Alberto Olmo, Sarath Sreedharan, and Subbarao Kambhampati. Large lan-  
649 guage models still can't plan (a benchmark for llms on planning and reasoning about change). In  
650 *NeurIPS 2022 Foundation Models for Decision Making Workshop*, 2022.

651  
652 Zihao Wang, Shaofei Cai, Guanzhou Chen, Anji Liu, Xiaojian Shawn Ma, and Yitao Liang. De-  
653 scribe, explain, plan and select: interactive planning with llms enables open-world multi-task  
654 agents. *Advances in Neural Information Processing Systems*, 36, 2024.

655 Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny  
656 Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in  
657 neural information processing systems*, 35:24824–24837, 2022.

658  
659 Xi Ye, Qiaochu Chen, Isil Dillig, and Greg Durrett. Satlm: Satisfiability-aided language models  
660 using declarative prompting. *Advances in Neural Information Processing Systems*, 36, 2024.

661  
662 Weihao Yu, Zihang Jiang, Yanfei Dong, and Jiashi Feng. Reclor: A reading comprehension dataset  
663 requiring logical reasoning. In *International Conference on Learning Representations*, 2019.

664 Yu Zhang, Hui-Ling Zhen, Zehua Pei, Yingzhao Lian, Lihao Yin, Mingxuan Yuan, and Bei Yu.  
665 Sola: Solver-layer adaption of llm for better logic reasoning. *arXiv preprint arXiv:2402.11903*,  
666 2024.

667 Wanjun Zhong, Siyuan Wang, Duyu Tang, Zenan Xu, Daya Guo, Jiahai Wang, Jian Yin, Ming  
668 Zhou, and Nan Duan. Ar-lsat: Investigating analytical reasoning of text. *arXiv preprint  
669 arXiv:2104.06598*, 2021.

## 671 672 A PROMPT FOR GENERATING BACKGROUND STORY

```
673
674 System:
675 You will be given a keyword and a name (can be a person's name or an animals' name). Your task
676 is to generate a background story with no more than 150 words according to the keyword about
677 this name. Your answer should be in JSON format with keys: category, story.
678
679 User:
680 keyword: swift
681 name: Jack
682
683 Assistant:
684 {
685   "category": "cat",
686   "story": "Jack was a sleek, agile cat known for his swift movements. Born in the bustling
687 city, he quickly learned to navigate the narrow alleyways and rooftops with ease.
688 His speed and agility made him a legend among the local feline community. Jack's
689 favorite pastime was chasing the elusive city birds, darting through the air with
690 unmatched precision. Despite his wild spirit, he had a soft spot for the elderly
691 woman who fed him daily. She named him Jack after the nimble character from her
692 favorite childhood story, believing that his swiftness was a sign of good fortune.
693 Jack's swift nature wasn't just physical; he had a quick wit and an uncanny ability
694 to sense danger, keeping him safe in the urban jungle. His legend grew, and he
695 became a symbol of agility and cleverness in the neighborhood."
696 }
697
698 User:
699 keyword: success
700 name: Amy
701
702 Assistant:
703 {
704   "category": "human",
705   "story": "Amy grew up in a small town where opportunities were scarce. Despite the odds, she
706 was determined to make something of herself. After excelling in high school, she
707 earned a scholarship to a prestigious university. There, she majored in computer
708 science and quickly became known for her innovative ideas. Upon graduation, Amy co-
709 founded a tech startup that focused on sustainable technology. Her company quickly
710 gained traction, securing major investments and partnerships. Amy's relentless
711 work ethic and visionary leadership turned her startup into a multi-million dollar
712 enterprise within five years. Today, she is celebrated not only for her business
713 acumen but also for her commitment to mentoring young entrepreneurs and giving back
714 to her community. Amy's story is a testament to the power of perseverance, vision,
715 and hard work."
716 }
```

702  
703  
704  
705  
706  
707  
708  
709  
710  
711  
712  
713  
714  
715  
716  
717  
718  
719  
720  
721  
722  
723  
724  
725  
726  
727  
728  
729  
730  
731  
732  
733  
734  
735  
736  
737  
738  
739  
740  
741  
742  
743  
744  
745  
746  
747  
748  
749  
750  
751  
752  
753  
754  
755

```

User:
keyword: loyal
name: Bella

Assistant:
{
  "category": "dog",
  "story": "Bella was a golden retriever with a heart as pure as gold. From the moment she was
adopted by the Thompson family, she became their steadfast guardian and companion.
Bella had an uncanny ability to sense when someone in the family needed comfort,
always there with a nuzzle or a wagging tail. Her loyalty was most evident when
young Timmy got lost in the woods behind their house. Bella tracked his scent
tirelessly, refusing to rest until she found him. She stayed by his side through
the cold night, keeping him warm and safe until help arrived. Bella's unwavering
loyalty and love made her not just a pet, but a cherished member of the
family."
}

User:
keyword: [KEYWORD]
name: [NAME]

```

## B PROMPTS FOR TRANSLATING LOGIC EXPRESSIONS INTO NATURAL LANGUAGE

We utilized three examples for each type of logical expression to guide the LLMs in translating them into natural language. An example of the prompt we used in our experiments is shown below.

```

System:
You will be provided a logic expression, a reference of the existed predicate in the logic
expression, and a background story. Your task is to replace the placeholders in the logic
expression with appropriate predicates (no more than 5 words) so that the provided logic
expression represent a real world common sense rule.

Do not use 'not'. Do not use the words in the forbidden list. The words that are similar
in meaning to the words in the forbidden list are also not allowed.

Your answer is not required to closely connected to the background story. You can use any
predicates you like as long as their length is less than 5.

Your answer should be in JSON format with the provided keys.

User:
background story:
Moriarty was a large, fluffy Maine Coon cat with a surprisingly warm and gentle nature.
Despite his intimidating size and the mysterious name inspired by a famous literary villain,
Moriarty was the epitome of warmth and comfort. He lived in a cozy bookstore run by Mrs.
Lindon, an elderly woman who had taken him in as a stray. Moriarty had a special talent for
sensing when a customer needed a bit of extra warmth, often curling up in their laps as they
read. His thick fur and gentle purring provided solace to many, especially during the cold
winter months. Moriarty's presence turned the bookstore into a beloved local haven, where
people came not just for the books, but for the warm, comforting presence of their favorite
feline friend.

reference: is_fluffy
is_fluffy(Moriarty)  $\rightarrow$  (warm_and_gentle(Moriarty)  $\oplus$  is_calming(Moriarty)): If Moriarty is
fluffy, then he is either warm and gentle or calming, but not both.
forbidden list: ['warm_and_gentle', 'provide_comfort', 'attract_customers', 'is_friendly',
'brings_joy', 'is_fluffy', 'is_calming']

logic expression:  $\forall x (F7(x) \rightarrow (is\_fluffy(x) \wedge F8(x)))$ 
keys: ['F7', 'F8', 'universal_rule', 'specific_rule']
Note: x belongs to cat.

Assistant:
{
  "F7": "playful",
  "F8": "love_attention",
  "universal_rule": "All playful cats are fluffy and love attention.",
  "specific_rule": "If Moriarty is playful, then he is both fluffy and loves attention."
}

-----
Another Two Examples

```

756  
757  
758  
759  
760  
761  
762  
763  
764  
765  
766  
767  
768  
769  
770  
771  
772  
773  
774  
775  
776  
777  
778  
779  
780  
781  
782  
783  
784  
785  
786  
787  
788  
789  
790  
791  
792  
793  
794  
795  
796  
797  
798  
799  
800  
801  
802  
803  
804  
805  
806  
807  
808  
809

**User:**

**background story:**

Colt was a brilliant botanist with a passion for rare plant species. He spent years studying the Avicenniaceae family, specifically the mangrove plants, which thrive in the harshest of coastal environments. Colt's groundbreaking research on these unique plants led to the development of innovative solutions for coastal conservation. He discovered that the Avicennia species possessed remarkable adaptability, allowing them to survive in areas with high salinity and limited oxygen. Colt's findings opened doors to new methods for restoring damaged ecosystems and protecting endangered species. As a leading expert in his field, Colt collaborated with international organizations to implement his research, helping to preserve the delicate balance of our planet's mangrove ecosystems. His dedication to the Avicenniaceae family left a lasting impact on the scientific community.

**reference:** contribute\_to\_environmental\_conservation

dedicated\_to\_research(Colt)  $\rightarrow$  (make\_groundbreaking\_discoveries(Colt)  $\wedge$

contribute\_to\_environmental\_conservation(Colt)): If Colt is dedicated to research, then he will make groundbreaking discoveries and contribute to environmental conservation.

**forbidden list:** ['dedicated\_to\_research', 'make\_groundbreaking\_discoveries', 'contribute\_to\_environmental\_conservation', 'pursue\_higher\_education', 'become\_leading\_expert']

**logic expression:**  $\forall x (F5(x) \rightarrow (contribute\_to\_environmental\_conservation(x) \wedge F6(x)))$

**keys:** ['F5', 'F6', 'universal\_rule', 'specific\_rule']

**Note:** x belongs to human.

## C [REVISION: POTENTIAL SHORTCUT PROBLEM]

To further investigate the potential use of shortcuts, we conducted an experiment by removing universal rules from 60 randomly selected instances in ProverGen. We evaluated GPT-4 and Llama-3.1-70B-Instruct on this *corrupted* dataset. If the models were relying heavily on inherent knowledge as shortcuts, their performance would remain roughly unaffected despite the absence of universal rules. However, the results in Table 6 showed a significant drop in performance, indicating that these models do not heavily rely on background knowledge to solve the problems.

Table 6: Comparative performance of GPT-4o and Llama-3.1-70B-Instruct on the original dataset and the corresponding corrupted version.

| Model                  | Prompting Strategy | Original | Corrupted | Acc $\Delta$ |
|------------------------|--------------------|----------|-----------|--------------|
| GPT-4o                 | Direct             | 58.33    | 43.33     | -15.99       |
| GPT-4o                 | CoT                | 68.33    | 45.00     | -23.33       |
| Llama-3.1-70B-Instruct | Direct             | 65.00    | 48.33     | -16.67       |
| Llama-3.1-70B-Instruct | CoT                | 65.00    | 53.33     | -11.67       |

## D EXAMPLES FROM THE THREE SUBSETS OF PROVERGEN

### ProverGen-Easy

**Context:**

For every elephant, the elephant is either domesticated or wild, but not both.  
Anakin is a domesticated elephant.

**Question:** Based on the above information, is the following statement true, false, or uncertain?  
Anakin is not a wild elephant.

**Options:**

- A) True
- B) False
- C) Uncertain

The correct option is: A

-----Proofs-----

-----Step1-----

810

fact1: Anakin is a domesticated elephant.  
rule: For every elephant, the elephant is either domesticated or wild, but not both.  
conclusion: Anakin is not a wild elephant.

811

812

Therefore, it is true that Anakin is not a wild elephant. The correct option is: A.

813

814

### ProverGen-Medium

815

816

#### Context:

Everyone who either practices waltz or masters tango (but not both) has good dance skills. Sawyer has charisma. Sawyer is a performer. If Sawyer has good dance skills and has charisma, then he is a successful dancer. Sawyer is either a musician or a poet, but not both. Alice is a poet. Sawyer is a poet. If Sawyer is a performer, then he is either skilled at dancing or a musician, but not both.

817

818

819

820

**Question:** Based on the above information, is the following statement true, false, or uncertain? Sawyer is a successful dancer.

821

822

#### Options:

- A) True
- B) False
- C) Uncertain

823

824

825

The correct option is: A

826

827

-----Proofs-----

828

829

-----Step1-----

830

fact1: Sawyer is a poet.  
rule: Sawyer is either a musician or a poet, but not both.  
conclusion: Sawyer is not a musician.

831

832

-----Step2-----

833

834

fact1: Sawyer is a performer.  
fact2: Sawyer is not a musician.  
rule: If Sawyer is a performer, then he is either skilled at dancing or a musician, but not both.  
conclusion: Sawyer has good dance skills.

835

836

837

-----Step3-----

838

839

fact1: Sawyer has good dance skills.  
fact2: Sawyer has charisma.  
rule: If Sawyer has good dance skills and has charisma, then he is a successful dancer.  
conclusion: Sawyer is a successful dancer.

840

841

842

Therefore, it is true that Sawyer is a successful dancer. The correct option is: A.

843

844

### ProverGen-Hard

845

846

#### Context:

Buster does not respond to calls. All dogs that listen to commands and behave properly are well-trained. Buster is either obedient or playful, but not necessarily both. Buster follows routines. Buster is either playful or loving (or both). If Buster either responds to calls or follows routines (but not both), then he listens to commands. All well-trained dogs are obedient and have good manners. Buster behaves properly.

847

848

849

850

**Question:** Based on the above information, is the following statement true, false, or uncertain? Buster is loving.

851

852

#### Options:

- A) True
- B) False
- C) Uncertain

853

854

855

The correct option is: B

856

857

-----Proofs-----

858

859

-----Step1-----

860

fact1: Buster does not respond to calls.  
fact2: Buster follows routines.  
rule: If Buster either responds to calls or follows routines (but not both), then he listens to commands.  
conclusion: Buster listens to commands.

861

862

863



```

864 -----Step2-----
865 fact1: Buster listens to commands
866 fact2: Buster behaves properly.
867 rule: All dogs that listen to commands and behave properly are well-trained.
868 conclusion: Buster is well-trained.
869 -----Step3-----
870 fact1: Buster is well-trained.
871 rule: All well-trained dogs are obedient and have good manners.
872 conclusion: Buster is obedient.
873 -----Step4-----
874 fact1: Buster is obedient.
875 rule: Buster is either obedient or playful, but not necessarily both.
876 conclusion: Buster is not playful.
877 -----Step5-----
878 fact1: Buster is not playful.
879 rule: Buster is either playful or loving (or both).
880 conclusion: Buster is loving.
881
882 Therefore, it is true that Buster is loving. The correct option is: A.

```

## 884 E EXAMPLES OF THE TWO PROMPTING METHODS

885  
886 The prompts of the two prompting strategies we used in our experiments are listed below. We adopt  
887 2-shot in-context learning for both Standard and CoT prompting strategies.

### 888 Standard Prompting

```

890 System:
891 Given a problem statement as contexts, the task is to answer a logical reasoning question.
892 Your answer should be in JSON format with key: answer.
893
894 Context:
895 Anyone who can solve problems and communicate effectively has practical skills. If Paola
896 appreciates beauty, then she values precision and enjoys manual work. Paola appreciates beauty.
897 Paola is either skilled in mathematics or has practical skills, but not both. If someone enjoys
898 manual work and values attention to detail, then they can craft with their hands. Paola is
899 either skilled in mathematics or artistically inclined, but surprisingly, she is actually both.
900 Paola values attention to detail. Anyone who crafts with their hands has practical skills.
901
902 Question: Based on the above information, is the following statement true, false, or uncertain?
903 Paola is artistically inclined.
904
905 Options:
906 A) True
907 B) False
908 C) Uncertain
909
910 The correct option is: {
911   "answer": "A"
912 }
913 -----
914 Another Example
915 -----
916 Context:
917 [[CONTEXT]]
918
919 Question: [[QUESTION]]
920
921 Options:
922 [[OPTIONS]]
923
924 The correct option is:

```

### 914 CoT Prompting

```

916 System:
917 Given a problem statement as contexts, the task is to answer a logical reasoning question.
918 Your answer should be in JSON format with keys: reasoning, answer.

```

918  
919  
920  
921  
922  
923  
924  
925  
926  
927  
928  
929  
930  
931  
932  
933  
934  
935  
936  
937  
938  
939  
940  
941  
942  
943  
944  
945  
946  
947  
948  
949  
950  
951  
952  
953  
954  
955  
956  
957  
958  
959  
960  
961  
962  
963  
964  
965  
966  
967  
968  
969  
970  
971

```

Context:
Anyone who can solve problems and communicate effectively has practical skills. If Paola
appreciates beauty, then she values precision and enjoys manual work. Paola appreciates beauty.
Paola is either skilled in mathematics or has practical skills, but not both. If someone enjoys
manual work and values attention to detail, then they can craft with their hands. Paola is
either skilled in mathematics or artistically inclined, but surprisingly, she is actually both.
Paola values attention to detail. Anyone who crafts with their hands has practical skills.

Question: Based on the above information, is the following statement true, false, or uncertain?
Paola is artistically inclined.

Options:
A) True
B) False
C) Uncertain

The correct option is: {
  "reasoning": "Paola appreciates beauty. If Paola appreciates beauty, then she values
precision and enjoys manual work. Paola enjoys manual work. Paola enjoys manual
work. Paola values attention to detail. If someone enjoys manual work and
values attention to detail, then they can craft with their hands. Paola crafts
with her hands. Paola crafts with her hands. Anyone who crafts with their hands
has practical skills. Paola has practical skills. Paola has practical skills.
Paola is either skilled in mathematics or has practical skills, but not both.
Paola is not skilled in mathematics. Paola is not skilled in mathematics. Paola
is either skilled in mathematics or artistically inclined, but surprisingly,
she is actually both. Paola is artistically inclined. Therefore, it is true
that Paola is artistically inclined. The correct option is: A.",
  "answer": "A"
}
-----
Another Example
-----
Context:
[[CONTEXT]]

Question: [[QUESTION]]

Options:
[[OPTIONS]]

The correct option is:

```

## F TRAINING DETAILS

To create the ProverGen train set, we started by generating 5,000 instances for each difficulty level: easy, medium, and hard. We then applied data augmentation techniques to expand the dataset. Specifically, for each problem, we broke down every step of its reasoning chain to form new questions. Additionally, we replaced the final conclusions with random facts to create problems whose answer is uncertain. This process expanded our generated data pool to 50,000 instances.

Our experiments show that, as the number of training instances increased, the model’s performance consistently improved on in-distribution datasets and two out-of-distribution datasets: ProofWriter and ProntoQA. However, performance on FOLIO fluctuated as the number of training instances increases. After further experimentation, we found that 5,000 is a sweet spot that reaches a good balance between in-distribution and out-of-distribution performance. Therefore, we sample 5,000 instances from the generated data pool as the final training set for finetuning experiments.

Since FOLIO lacks reasoning chains, we could not apply the same augmentation techniques on it. We used the full set of FOLIO, consisting of 1,001 instances. For ProofWriter, which includes multiple questions per context similar to our data augmentation, we did not apply further augmentation.

We also finetuned the model on the full set of ProofWriter, which has 45,290 instances. The performance is comparable compared to the performance of model trained on a sampled subset consisting of 5,000 instances, indicating that the model may already be saturated at this size for an 8B parameter model. As a result, we report the results of the sampled subset.

For all finetuning experiments, we experimented with different training hyper-parameters, such as training epochs, learning rate and training set size, and selected the best configurations on held-out validation sets.

## G [REVISION: EXPERIMENTING WITH 5-SHOT EXAMPLES]

To further explore the impact of few-shot examples, we conducted additional experiments using 5-shot prompts. The results are presented in Table 7. It can be concluded that incorporating more examples (5-shot vs. 2-shot) into the prompt did not consistently improve the performance of LLMs. This inconsistency may stem from several factors, including variations in the models’ in-context learning capabilities.

Table 7: The performance of LLMs in the 5-shot setting. The values in parentheses indicate the change in performance relative to the 2-shot setting.

|                           | ProverGen         |                  |                  | ProntoQA         | ProofWriter      | FOLIO            | Avg $\Delta$ |
|---------------------------|-------------------|------------------|------------------|------------------|------------------|------------------|--------------|
|                           | Easy              | Medium           | Hard             |                  |                  |                  |              |
| <i>Standard Prompting</i> |                   |                  |                  |                  |                  |                  |              |
| GPT-4o                    | 85.20<br>(-2.00)  | 68.40<br>(-0.20) | 43.80<br>(-2.40) | 88.20<br>(-3.60) | 57.33<br>(+1.00) | 72.86<br>(+5.00) | -0.37        |
| Claude-3.5-Sonnet         | 93.80<br>(+8.20)  | 76.80<br>(+8.60) | 51.00<br>(+8.20) | 85.20<br>(-3.40) | 50.33<br>(-4.67) | 74.29<br>(-3.56) | +2.23        |
| Llama3.1-8B-Instruct      | 64.40<br>(+17.80) | 41.80<br>(-1.20) | 38.00<br>(-1.00) | 58.00<br>(+7.60) | 44.50<br>(+0.70) | 46.43<br>(-7.86) | +2.67        |
| Llama3.1-70B-Instruct     | 82.80<br>(+0.80)  | 64.40<br>(+0.20) | 49.20<br>(+1.60) | 74.00<br>(-6.60) | 51.33<br>(+1.00) | 69.29<br>(+1.43) | -0.26        |
| Mistral-7B-Instruct       | 57.40<br>(+0.60)  | 48.40<br>(+1.60) | 37.40<br>(+0.20) | 52.40<br>(+2.40) | 47.83<br>(+5.50) | 52.86<br>(-1.43) | +1.48        |
| Mistral-Large-Instruct    | 85.00<br>(+0.40)  | 71.40<br>(+2.20) | 53.60<br>(+4.00) | 67.40<br>(-3.60) | 63.50<br>(+3.17) | 76.43<br>(-0.71) | +0.91        |
| Mixtral-8x22B-Instruct    | 77.60<br>(+2.20)  | 58.00<br>(+0.60) | 39.60<br>(+0.60) | 60.00<br>(-5.20) | 40.50<br>(+0.33) | 73.57<br>(-0.72) | -0.37        |
| <i>CoT Prompting</i>      |                   |                  |                  |                  |                  |                  |              |
| GPT-4o                    | 95.00<br>(+0.80)  | 81.20<br>(+1.80) | 55.00<br>(+5.00) | 99.20<br>(-0.80) | 71.00<br>(+3.67) | 74.29<br>(+2.15) | +2.10        |
| Claude-3.5-Sonnet         | 92.00<br>(-3.20)  | 81.60<br>(-2.00) | 61.00<br>(+4.60) | 95.80<br>(-3.40) | 75.50<br>(-0.83) | 82.86<br>(+2.15) | -0.45        |
| Llama3.1-8B-Instruct      | 80.60<br>(+5.00)  | 45.40<br>(-1.20) | 31.20<br>(-2.40) | 74.80<br>(-4.80) | 58.67<br>(+1.84) | 57.86<br>(-5.68) | -1.21        |
| Llama3.1-70B-Instruct     | 92.60<br>(+2.20)  | 73.40<br>(+0.20) | 50.20<br>(+3.40) | 92.80<br>(-2.60) | 67.00<br>(-4.17) | 75.00<br>(+0.71) | -0.04        |
| Mistral-7B-Instruct       | 65.60<br>(-6.40)  | 50.20<br>(-0.80) | 38.00<br>(-3.80) | 65.80<br>(+4.60) | 48.33<br>(+2.33) | 63.57<br>(-0.01) | -0.68        |
| Mistral-Large-Instruct    | 94.40<br>(+1.80)  | 77.60<br>(+1.80) | 56.60<br>(+4.40) | 99.00<br>(+0.40) | 77.67<br>(+4.17) | 82.86<br>(-0.71) | +1.98        |
| Mixtral-8x22B-Instruct    | 91.00<br>(+3.40)  | 73.80<br>(+7.00) | 50.20<br>(+2.60) | 86.20<br>(+6.60) | 59.67<br>(+2.00) | 72.14<br>(-1.43) | +3.36        |

## H [REVISION: QUALITY CONTROL]

The ProverGen framework employs three heuristic methods to ensure the quality of the generated data, addressing different aspects of data integrity and consistency.

- **Logic Validation:** Our framework includes a robust logic validation step. For each instance, we input both core premises and distractions into the symbolic prover to ensure they correctly deduce the conclusion’s truth value.
- **Conflicts Resolution:** During rule translation, we check for previously used predicates to avoid redundancy and potential conflicting facts. Additionally, we utilize LLMs to assess

1026 whether the generated universal rules align with real-world knowledge. In cases of conflict,  
1027 we opt for specific rules instead.

1028 • **Translation Quality Control:** We apply a heuristic method to ensure that all involved  
1029 entities appear in both symbolic expression and natural language expression. For example,  
1030 when translating "poet(Sawyer)" into "Sawyer is a poet", we verify that both the name  
1031 "Sawyer" and the predicate "poet" are present in the translation.  
1032

1033 Additionally, the training part of the paper also serves as a quality-checking process. Finetuning  
1034 on the generated dataset enhances the performance of LLMs on both in-distribution and Out-of-  
1035 Distribution datasets, indicating the relatively high quality of the generated data.

1036 To further investigate the quality of the generated data, we performed a manual evaluation of 60  
1037 randomly sampled examples (20 from each subset) from our dataset. The evaluation revealed no  
1038 translation errors or indirect contradictions, suggesting that the likelihood of potential errors is less  
1039 than 2%.  
1040

1041  
1042  
1043  
1044  
1045  
1046  
1047  
1048  
1049  
1050  
1051  
1052  
1053  
1054  
1055  
1056  
1057  
1058  
1059  
1060  
1061  
1062  
1063  
1064  
1065  
1066  
1067  
1068  
1069  
1070  
1071  
1072  
1073  
1074  
1075  
1076  
1077  
1078  
1079