LogicTree: Improving Complex Reasoning of LLMs via Instantiated Multi-step Synthetic Logical Data

Zehao Wang[1]*, **Lin Yang**[2], **Jie Wang**[1]†, **Kehan Wang**[1], **Hanzhu Chen**[1], **Bin Wang**[2]
Jianye Hao[2,3], **Defu Lian**[1], **Bin Li**[1], **Enhong Chen**[1]
[1]MoE Key Laboratory of Brain Science and Education, Psychological and Cognition,
University of Science and Technology of China
[2]Noah's Ark Lab, Huawei Technologies, [3] Tianjin University

Abstract

Despite their remarkable performance on various tasks, Large Language Models (LLMs) still struggle with logical reasoning, particularly in complex and multi-step reasoning processes. Among various efforts to enhance LLMs' reasoning capabilities, synthesizing large-scale, high-quality logical reasoning datasets has emerged as a promising direction. However, existing methods often rely on pre-defined templates for logical reasoning data generation, limiting their adaptability to real-world scenarios. To address the limitation, we propose **LogicTree**, a novel framework for efficiently synthesizing multi-step logical reasoning dataset that excels in both complexity and instantiation. By iteratively searching for applicable logic rules based on structural pattern matching to perform backward deduction, **LogicTree** constructs multi-step logic trees that capture complex reasoning patterns. Furthermore, we employ a two-stage LLM-based approach to instantiate various real-world scenarios for each logic tree, generating consistent real-world reasoning processes that carry contextual significance. This helps LLMs develop generalizable logical reasoning abilities across diverse scenarios rather than merely memorizing templates. Experiments on multiple benchmarks demonstrate that our approach achieves an average improvement of 9.4% in accuracy for LLMs on complex logical reasoning tasks.

1 Introduction

Logical reasoning is a crucial capability for large language models (LLMs) [15, 24, 14], providing substantial support for complex tasks such as coding, mathematics, and other higher-order cognitive abilities [30, 28, 45]. Recent advancements in the reasoning capabilities of LLMs have been impressive[46], with one of the key progress attributed to the availability of high-quality reasoning data[18, 9, 50, 6]. Despite these strides, LLMs still encounter challenges in complex multi-step logical reasoning tasks[4, 31, 42], underscoring a significant scarcity of high-quality multi-step logical reasoning datasets.

Early logical reasoning datasets are primarily constructed through manual data collection and annotation [19, 55, 13]. While these datasets are often diverse and intricate, their creation is labor-intensive and time-consuming for further training. In recent years, several studies have explored data synthesis approaches to generate logical reasoning data [37]. Some studies have employed a set of atomic logic rules to evaluate and enhance the formal reasoning abilities of LLMs[41, 33]. Similarly, other research has synthetically created multi-step logical reasoning datasets by leveraging natural language templates[30, 29, 7, 2]. Furthermore, there is an increasing interest in generating reasoning processes

*This word was done when Zehao Wang interned at Huawei. Email: zh-wang@mail.ustc.edu.cn
†Corresponding author. Email:jiewangx@ustc.edu.cn

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

for existing complex logical reasoning questions[21, 39]. However, existing data synthesis methods typically exhibit the following limitations: (1) insufficient complexity, as they typically generate simplistic reasoning patterns with limited types of reasoning rules and shallow reasoning steps; and (2) inadequate real-world scenario instantiation, often resulting from the combination of entities that lack contextual or conceptual relevance, which has been shown to weaken the robustness and generalization capabilities of LLMs in reasoning tasks[51, 47]. Real-world scenario instantiation refers to the process of grounding synthetic reasoning data in concrete and context-rich real-world scenarios that incorporate semantically relevant entities or events, capable of reflecting the complexity and diversity of real-world reasoning tasks.

To address these challenges, we propose **LogicTree**, a novel framework for efficiently synthesizing multi-step logical reasoning dataset, offering significant advantages in terms of **complexity** and **instantiation**. Our method leverages first-order logic rules to generate complex multi-step logical reasoning trees, which are then instantiated using LLMs to produce natural language reasoning questions with real-world scenarios and reasoning processes that carry contextual significance. The logic trees generated by **LogicTree** exhibit complex reasoning patterns (as presented in Tables 10 and 11), facilitating the improvement of advanced reasoning abilities in LLMs for tackling complex tasks (Sec.4.3). Moreover, contextually diverse instantiated reasoning data help LLMs develop generalizable logical reasoning skills, rather than simply memorizing implicit relationships between facts[24, 51, 40].

Figure 1 presents an overview of **LogicTree**, illustrating the data synthesis process in detail. Firstly, we construct symbolic logical reasoning trees via a backward deduction procedure, where leaf nodes are iteratively expanded by applying formal logic rules identified through structural pattern matching of formulas. In this way, we generate multi-step logical reasoning trees that incorporate diverse and complex reasoning patterns, encompassing both propositional logic and first-order logic. Secondly, we introduce a **two-stage LLM-based approach** to instantiate logical reasoning trees: (1) assigning contextually relevant entities or events to the logical symbols in the leaf nodes to construct realistic reasoning scenarios; and (2) sequentially translating all intermediate nodes in the logical tree to generate step-by-step natural language reasoning processes. By controlling the thematic domains of the content generated by LLMs, we instantiate each logical reasoning tree with multiple diverse scenarios, thereby enhancing the overall diversity and contextual richness of the dataset (see Figures 4 and 3). Finally, we apply post-processing to the synthesized data to verify the logical consistency of the instantiated content and construct logical reasoning instances, each consisting of a set of premises, a conclusion–answer pair, and the corresponding reasoning process. Experiments conducted across a wide range of LLMs demonstrate that the instantiated complex logical reasoning data synthesized by LogicTree effectively enhances the models' reasoning capabilities. Furthermore, our analytical experiments (Table 3) indicate that diverse instantiation scenarios are beneficial for improving the generalization of reasoning abilities.

The main contributions of this work include:

- We introduce **LogicTree**, a novel framework for synthesizing multi-step logical reasoning datasets by generating complex logic trees with first-order logic rules and instantiating them into realistic reasoning scenarios, enabling LLMs to develop advanced and generalizable logical reasoning abilities (Sec.3).
- We present a **two-stage LLM-based instantiation technique** that injects realistic statements into the symbolic reasoning trees, generating coherent and contextually grounded natural language reasoning processes (Sec.3).
- Experimental results demonstrate that the synthesized dataset significantly enhance the logical reasoning performance of LLMs across several challenging benchmarks, leading to an average accuracy improvement of up to 9.4% (Sec.4).

2 Preliminary

Logical reasoning is a cognitive process that involves deriving valid conclusions from given premises based on formal **logic rules** [15, 27, 5]. It is a cornerstone of fields like artificial intelligence, mathematics, and philosophy, enabling systematic decision-making and problem-solving[24, 34]. The main formal logic systems include **propositional logic** and **first-order logic(FOL)**. Propositional

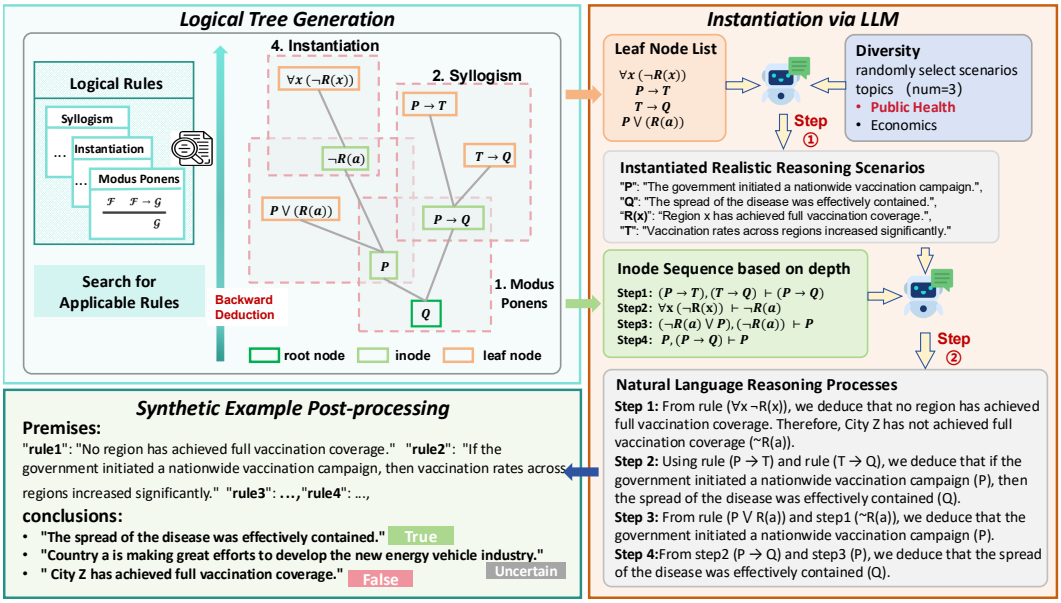

Figure 1: An overview of LogicTree, which comprises three key modules: (1) Logical reasoning tree generation via iterative backward deduction based on structural pattern matching; (2) Reasoning scenario instantiation using a two-stage LLM-based approach; (3) Synthetic reasoning example post-processing.

logic focuses on propositions or statements that are either true or false, employing logical operators such as $\wedge$ (and), $\vee$ (or), $\neg$ (not), and $\rightarrow$ (implies) to connect these statements. In contrast, FOL extends propositional logic by handling more complex statements involving variables, quantifiers (e.g., $\forall$ for "for all" and $\exists$ for "there exists"), and predicates.

In a formal logic system, a logical expression composed of the aforementioned symbols is referred to as a **formula** (e.g., $A \rightarrow B$, $\forall x P(x)$). If an expression contains no logical operators, it is called an **atomic formula**. By representing premises or statements as formulas, logical rules can be systematically applied to derive new conclusions. For instance, *Universal Instantiation* and *Hypothetical Syllogism* are two well-known inference rules:

$$\text{UI:} \quad \forall x(p(x)) \vdash p(a) \tag{1}$$

$$\text{HS:} \quad \big((p \rightarrow q) \wedge (q \rightarrow r)\big) \vdash (p \rightarrow r) \tag{2}$$

Here, $\vdash$ denotes "derives", meaning that the formulas preceding it serve as premises, while the formula following it represents the conclusion. The *Hypothetical Syllogism (HS)* rule states that if "p implies q" and "q implies r", then we can conclude that "p implies r". For additional logical rules and further details, please refer to Appendix C.

3 Method

In this section, we introduce the design and implementation of **LogicTree**, a novel framework for synthesizing instantiated complex logical reasoning datasets. Figure 1 overviews the workflow of our approach, which consists of three main modules: **logical reasoning tree generation**, **reasoning scenario instantiation** and **synthetic reasoning example post-processing**. First, based on structural pattern matching, we construct symbolic logical reasoning trees through a backward deduction process(Sec. 3.1). Next, we leverage LLMs to instantiate the logical reasoning trees with real-world scenarios(Sec.3.2). Finally, we verify the logical consistency of the instantiated content and generate synthetic logical reasoning instances(Sec.3.3).

3.1 Logical Reasoning Tree Generation

Logical Reasoning Tree Each node in a logical reasoning tree (logic tree) corresponds to a logical expression (formula) that can be derived from its child nodes based on logic rules. The role of each node varies depending on its position within the tree: ***leaf nodes*** represent premises, the ***root node*** denotes the final conclusion, and ***intermediate nodes (inodes)*** encapsulate individual reasoning steps.

In order to construct logical reasoning tree,we designed a backward deduction method based on structural pattern matching of formulas, which starts from a given formula and searches for applicable logic rules to derive the premises that support it. When determining whether a logic rule can be applied to a formula, this method does not require the formula to be identical to the conclusion of the logic rule; instead, it only compares whether the structural patterns of their are matched. In this way, we generate multi-step logical reasoning trees that incorporate diverse and complex reasoning patterns, encompassing both propositional logic and first-order logic. We provide additional detailed information in the Appendix C.2, including the types of logical rules involved and the complexity of the logical trees (e.g., depth, breadth, and the number of rules).

Algorithm 1 Logical Reasoning Tree Generation Process

Input: Root conclusion formula $\mathcal{Q}$, number of iterations N, rules list $\mathcal{R} = \{r_1, r_2, \ldots, r_m\}$
$\mathcal{T} = init\_logictree(\mathcal{Q})$
$k = 1$
while $k \leq N$ **do**
  $\mathcal{L} = random\_select\_leaf\_node(\mathcal{T})$
  $\mathcal{A} = \emptyset$ *#appliable rules list*
  for $r \in \mathcal{R}$ **do**
    if $is\_appliable\_rule(\mathcal{L}.AST, r.conclusion.AST)$ **then**
      $\mathcal{A} \leftarrow \mathcal{A} \cup \{r\}$
    end if
  end for
  $appliable\_rule = random\_choice(\mathcal{A})$
  $\mathcal{L}.childs = backward\_deduction(\mathcal{L}, appliable\_rule)$
  $k = k + 1$
end while
Output: Final logical reasoning tree $\mathcal{T}$

Structural Pattern Matching of Formulas Logical formulas can be represented as *Abstract Syntax Trees (ASTs)*. In this AST representation, the root node typically corresponds to the logical operator with the highest precedence. The operands of this operator are then represented by the subtrees branching from the root. This decomposition process is applied recursively to the subtrees, continuing until the tree's nodes represent only atomic formulas, which form the leaves of the tree. ***The structure of a formula is considered to match the structural pattern of another if their AST's respective operator skeletons are isomorphic.*** This means their ASTs have the same upper-level structure. For example, consider the formulas $(A \vee B) \rightarrow (\neg C)$ and $F \rightarrow G$. Although the two formulas are not identical, they have the same outermost logical operator, namely $\rightarrow$. Since $F \rightarrow G$ are the conclusions of Rule (2), we can apply this rule to $(A \vee B) \rightarrow (\neg C)$ for backward deduction, resulting in the premises $(A \vee B) \rightarrow G$ and $G \rightarrow (\neg C)$. Therefore, if the conclusion formula of a rule matches the structural pattern of the target formula, the rule can be applied to the target formula. Refer to Appendix C.1 and Figure 5 for more details.

Multi-step Backward Deduction Generation As shown in Algorithm 1, our logical reasoning tree generation follows a backward deduction process, iteratively searching for applicable formal logical rules based on structural pattern matching to decompose existing formula nodes into new ones. Initially, the logical tree consists of a single root node, which is a randomly generated formula. Then, the process iterates as follows until the desired number of iterations is reached: Randomly select a leaf node in the tree, apply relevant rules to perform backward deduction, and derive the necessary premises, which are added as child nodes to the selected formula. The number of iterations and the search strategy can control the structure of the final logical reasoning tree, specifically its depth and width. Examples of logical reasoning trees can be found in the Appendix E.

3.2 Reasoning Scenario Instantiation via LLMs

Previous works[30, 29, 3] have employed template based approach to translate the symbolic formulas into natural language statements. However, these methods fail to capture the contextual semantics of logical relationships between formulas, often generating statements with limited real-world applicability.

Thus, we aim to instantiate logical reasoning trees with realistic reasoning scenarios, which involves translating symbolic logical expressions into concrete statements that carry contextual significance. Given the advanced text generation capabilities of modern LLMs (e.g., GPT-4 [1]), an intuitive approach is to leverage LLMs for instantiation. However, due to the inherent limitations of LLMs in handling complex logical reasoning, directly prompting them to generate realistic logical reasoning problems from symbolic logic trees may introduce errors. To mitigate this issue, we design the following two-state prompting strategy to guide LLMs in instantiating the logical reasoning tree. The details of prompts are provided in Appendix D.

(1) **Logical tree instantiation**: We extract all the formulas that are leaf nodes in the logical reasoning tree and input them into LLMs to instantiate a real-world reasoning scenario. During this process, LLMs assign real-world statements to atomic formulas while ensuring consistency with their predefined logical relationships. Once all leaf nodes are translated into natural language, a coherent and contextually grounded logical reasoning scenario is formed.

(2) **Reasoning process translation**: We obtain a complete, step-by-step symbolic reasoning chain by sorting the internal nodes of the logical tree in descending order of depth. At each step, an inode is derived from its child nodes according to the corresponding logical rules. We then leverage LLMs to translate the symbolic reasoning steps into a natural language reasoning process within the previously instantiated scenario.

In this process, **the LLM is guided by a correct and rigorous symbolic reasoning skeleton to generate a step-by-step, instantiated natural language reasoning process**. This approach significantly reduces generation errors, as the LLM is not required to perform the reasoning autonomously; it only needs to understand the logical relationships and perform symbolic translatio, which is well within the capabilities of LLMs[32, 36]. Furthermore, by **controlling the themes of the reasoning scenarios used by the LLM**, we can ensure diversity in the synthesized data (Figure 4). This approach enables the generation of coherent, step-by-step logical reasoning processes in natural language that incorporate a wide range of realistic scenarios.

3.3 Synthetic Reasoning Example Post-processing

We conduct a systematic verification of the logical consistency of the instantiated content to filter out erroneous data. Specifically, this involves prompting the LLM to output the instantiated natural language statements paired with their corresponding logical expressions, typically in the format "statement [expression]". We then perform an exact string comparison between the logical expression provided by the LLM and the original source logical expression to identify any inconsistencies. The entire natural language reasoning process is validated once all statements are confirmed to be logically consistent with their symbolic expressions, as the overall correctness is inherently guaranteed by the logic tree. This rigorous verification process enables us to filter out the data identified as erroneous, thereby significantly improving the overall data quality.

Then, we utilize the verified instantiated content to construct logical reasoning data instances. A logical reasoning instance is composed of a set of premises, a conclusion-answer pair, and the corresponding reasoning process. In particular, the natural language statements of the leaf nodes are combined to form the premises. Then we can construct conclusions with different types of answers ('True', 'False' or 'Uncertain') based on the logical tree. For answer label "True", we use the statement of root node as the conclusion. For answer label "False", we use the negated statement of the root node as the conclusion. For answer label "Uncertain", we introduce some irrelevant statements as distractors. Examples of synthetic logical reasoning data can be found in Appendix E.

4 Experiments

To evaluate the effectiveness of our **LogicTree**, we design a suite of experiments that not only demonstrate its significant impact on enhancing LLMs' reasoning abilities, but also provide in-depth analytical insights. Specifically, we divide the experiments into three parts:

- To comprehensively assess the enhancements of our approach in improving the logical reasoning abilities of LLMs, we conduct evaluations across multiple benchmarks and compare it against several other synthetic logical datasets (Sec.4.2).
- To further evaluate the model's multi-step logical reasoning ability, we assess **LogicTree** and baselines on Multi-LogiEval, which categorizes questions by reasoning steps from one to five (Sec.4.3).
- To analyze the key factors contributing to LogicTree's effectiveness, we conduct ablation studies on three core components: (1) instantiating real-world reasoning scenarios, (2) incorporating natural language reasoning processes under instantiated scenarios, and (3) ensuring diversity in instantiated scenarios (Sec.4.5).
- To further evaluate the generalization enhancement of model reasoning capabilities by LogicTree, we conduct additional experiments across a diverse range of task types, covering domains such as logic, math, code, NLI, and others (Sec.4.4).
- To assess the generalizability of our synthetic data, we extended our evaluation to include models from different families and scales, specifically Qwen2.5-1.5B/3B and the Deepseek-R1-Distill series (Appendix.B.3).

4.1 Experimental Setup

We briefly explain the experimental settings. Refer to Appendix B for more details.

Syntheised Training Dataset. We generated 5,000 symbolic logic trees with depths from 2 to 15, and instantiated each into 3 semantically diverse scenarios, yielding 15,000 reasoning problems. After applying an automatic filtering process that discarded 8.73% of noisy or invalid samples, the final dataset for LLM training contains 13.8k high-quality, multi-step reasoning instances.

Evaluation Benchmarks. To evaluate the effectiveness of our proposed **LogicTree**, we consider a diverse set of reasoning tasks and datasets that require rigorous logical reasoning: (a) **LogicBench**[33]: a novel task designed to comprehensively evaluate the model's performance on each inference rule. (b) **LogiQA2.0**[19]: A collection of challenging real-world logical reasoning problems from civil service entrance exams. (c) Three **BIG-Bench Hard (BBH)**[38] tasks of varying difficulty levels: logical deduction with three, five, and seven objects. (d) **FOLIO**[13]: An expert-written, logically complex dataset for first-order logic reasoning. (e) Two **AGIEval**[55] tasks focused on logical reasoning: LAST-AR and LAST-LR. (f) **Multi-LogiEval**[35]: A comprehensive dataset incorporating varying reasoning depths for logical complexity.

Baselines. To demonstrate the superiority of our method, we evaluated the following baselines. **(i) Vanilla**: standalone LLMs without additional training. Vanilla LLMs represent the original capabilities of LLMs. **(ii) PARARULE**[2]: it generates deductive processes by repeatedly applying deduction rules to a given set of facts and transforms these processes into sentences based on natural language templates. **(iii) LogicAsker**[41]: a framework to evaluate the ability of LLMs to handle individual atomic logical reasoning rules and generate targeted samples for improvement. **(iv) FLD$_{\times 2}$**[29]: it introduces systematic design principles for logical data synthesis and randomly combines various rules to construct multi-step reasoning datasets.

Models and Training Settings. To rigorously validate the effectiveness of LogicTree, we conduct extensive experiments using a diverse set of prominent open-source models spanning various families and scales. We utilize models from the Llama-3.1, Mistral-v0.3, Qwen2.5, and Deepseek-R1-Distill families, with parameter scales ranging from 1.5B to 70B. We employ two distinct fine-tuning strategies: full fine-tuning is applied to smaller-scale models (under 8B), while the larger 70B model utilizes LoRA fine-tuning.

Table 1: Main results on five types of reasoning tasks. $\triangle$ means the margin between Vanilla and training by LogicTree. "Avg" means the average accuracy across five benchmarks for each methods. We bold the best results for each LLM backbone and underline the second-best results.

Model	LogicBench	LogiQA2.0	FOLIO	BBH-Logic	AGIEval		Avg.
					LR	AR	
Llama-3.1-8B							
vanilla	$80.0_{\pm 0.4}$	$42.4_{\pm 0.2}$	$50.8_{\pm 0.4}$	$39.3_{\pm 0.3}$	$48.4_{\pm 0.3}$	$20.7_{\pm 0.1}$	46.9
+ PARARULE	$84.4_{\pm 0.3}$	$53.3_{\pm 0.2}$	$50.3_{\pm 0.3}$	$42.4_{\pm 0.1}$	$\underline{52.9}_{\pm 0.2}$	$21.7_{\pm 0.2}$	50.8
+ LogicAsker	$\underline{85.0}_{\pm 0.3}$	$\mathbf{54.7}_{\pm 0.2}$	$\underline{54.6}_{\pm 0.2}$	$43.7_{\pm 0.1}$	$51.2_{\pm 0.2}$	$\underline{23.1}_{\pm 0.1}$	52.0
+ FLD$_{\times 2}$	$83.6_{\pm 0.2}$	$53.3_{\pm 0.2}$	$54.5_{\pm 0.3}$	$39.5_{\pm 0.2}$	$51.8_{\pm 0.3}$	$21.3_{\pm 0.1}$	50.7
+ LogicTree	$\mathbf{90.6}_{\pm 0.2}$	$\underline{53.5}_{\pm 0.1}$	$\mathbf{57.6}_{\pm 0.2}$	$\mathbf{53.2}_{\pm 0.2}$	$\mathbf{56.1}_{\pm 0.1}$	$\mathbf{26.5}_{\pm 0.2}$	**56.3**
Δ (**Relative Gain**)	+10.6 (+13.3%)	+11.1 (+26.2%)	+6.8 (+13.4%)	+13.9 (+35.4%)	+7.7 (+15.9%)	+5.8 (+28.0%)	+9.4 (+20.0%)
Qwen-2.5-7B							
vanilla	$83.1_{\pm 0.3}$	$63.8_{\pm 0.4}$	$54.3_{\pm 0.4}$	$72.8_{\pm 0.3}$	$65.3_{\pm 0.3}$	$23.9_{\pm 0.2}$	60.5
+ PARARULE	$82.9_{\pm 0.3}$	$65.1_{\pm 0.1}$	$55.3_{\pm 0.2}$	$72.8_{\pm 0.4}$	$\underline{68.3}_{\pm 0.2}$	$24.5_{\pm 0.1}$	61.5
+ LogicAsker	$\underline{84.3}_{\pm 0.3}$	$\mathbf{64.9}_{\pm 0.1}$	$\underline{58.7}_{\pm 0.2}$	$72.8_{\pm 0.2}$	$66.7_{\pm 0.2}$	$\underline{24.5}_{\pm 0.1}$	62.0
+ FLD$_{\times 2}$	$83.4_{\pm 0.1}$	$65.1_{\pm 0.3}$	$59.6_{\pm 0.3}$	$73.6_{\pm 0.2}$	$67.3_{\pm 0.1}$	$26.1_{\pm 0.2}$	62.5
+ LogicTree	$\mathbf{89.2}_{\pm 0.1}$	$\underline{64.9}_{\pm 0.1}$	$\mathbf{63.8}_{\pm 0.1}$	$\mathbf{74.3}_{\pm 0.3}$	$\mathbf{69.3}_{\pm 0.1}$	$\mathbf{28.7}_{\pm 0.2}$	**65.0**
Δ (**Relative Gain**)	+6.1 (+7.3%)	+1.1 (+1.7%)	+9.5 (+17.5%)	+1.5 (+2.1%)	+4.0 (+6.1%)	+4.8 (+20.1%)	+4.5 (+7.4%)
Mistral-7B-v0.3							
vanilla	$77.5_{\pm 0.4}$	$49.7_{\pm 0.3}$	$51.2_{\pm 0.2}$	$45.5_{\pm 0.3}$	$49.9_{\pm 0.3}$	$20.4_{\pm 0.1}$	49.0
+ PARARULE	$\underline{78.8}_{\pm 0.2}$	$\underline{51.2}_{\pm 0.3}$	$50.3_{\pm 0.3}$	$47.2_{\pm 0.2}$	$50.4_{\pm 0.2}$	$22.6_{\pm 0.2}$	50.1
+ LogicAsker	$78.8_{\pm 0.3}$	$50.6_{\pm 0.2}$	$\mathbf{56.7}_{\pm 0.1}$	$46.8_{\pm 0.1}$	$50.2_{\pm 0.2}$	$\underline{23.1}_{\pm 0.1}$	51.0
+ FLD$_{\times 2}$	$78.0_{\pm 0.2}$	$50.6_{\pm 0.2}$	$\underline{54.5}_{\pm 0.3}$	$\underline{48.1}_{\pm 0.2}$	$\underline{50.4}_{\pm 0.3}$	$22.2_{\pm 0.1}$	50.6
+ LogicTree	$\mathbf{81.9}_{\pm 0.2}$	$\mathbf{52.3}_{\pm 0.1}$	$52.5_{\pm 0.2}$	$\mathbf{48.7}_{\pm 0.2}$	$\mathbf{51.4}_{\pm 0.1}$	$\mathbf{25.7}_{\pm 0.2}$	**52.1**
Δ (**Relative Gain**)	+4.4 (+5.7%)	+2.6 (+5.2%)	+1.3 (+2.5%)	+3.2 (+7.0%)	+1.5 (+3.0%)	+5.3 (+26.0%)	+3.1 (+6.3%)
Llama-3.1-70B							
vanilla	$91.3_{\pm 0.2}$	$67.1_{\pm 0.2}$	$59.6_{\pm 0.4}$	$75.3_{\pm 0.2}$	$82.2_{\pm 0.1}$	$28.3_{\pm 0.1}$	67.3
+ PARARULE	$92.5_{\pm 0.3}$	$68.2_{\pm 0.2}$	$61.1_{\pm 0.3}$	$75.6_{\pm 0.1}$	$82.4_{\pm 0.2}$	$26.1_{\pm 0.2}$	67.7
+ LogicAsker	$93.1_{\pm 0.3}$	$69.9_{\pm 0.2}$	$\underline{63.4}_{\pm 0.2}$	$75.2_{\pm 0.1}$	$82.8_{\pm 0.2}$	$\underline{30.8}_{\pm 0.1}$	69.2
+ FLD$_{\times 2}$	$\underline{93.7}_{\pm 0.2}$	$\underline{69.9}_{\pm 0.1}$	$63.1_{\pm 0.2}$	$74.8_{\pm 0.3}$	$\underline{83.1}_{\pm 0.3}$	$26.1_{\pm 0.1}$	68.5
+ LogicTree	$\mathbf{94.4}_{\pm 0.2}$	$\mathbf{70.3}_{\pm 0.1}$	$\mathbf{65.2}_{\pm 0.2}$	$\mathbf{76.0}_{\pm 0.0}$	$\mathbf{83.7}_{\pm 0.2}$	$\mathbf{31.3}_{\pm 0.1}$	**70.2**
Δ (**Relative Gain**)	+3.1 (+3.4%)	+3.2 (+4.8%)	+5.6 (+9.4%)	+0.7 (+0.9%)	+1.5 (+1.8%)	+3.0 (+10.6%)	+2.9 (+4.3%)

4.2 Main Results

To validate the significant improvement in logical reasoning abilities enabled by our synthesized data, we evaluate **LogicTree** on multiple downstream logical reasoning tasks. As highlighted in Table 1, **LogicTree** shows consistent and substantial performance improvements across all backbone models and most benchmarks against the baselines, demonstrating the effectiveness of instantiating complex logical reasoning trees. Specifically, LogicTree surpasses vanilla models on all benchmarks, achieving significant margins. On LogicBench and AGIEval-LR, it improves LLama-3.1-8B by 10.6% and 7.7%, respectively. Even on AGIEval-AR, a particularly challenging benchmark for LLMs, our approach still achieves a 6% improvement. Meanwhile, our method achieves average improvements of 9.4%, 3.1%, and 2.9% on Llama-3.1-8B, Mistral-7B-v0.3, and Llama-3.1-70B, respectively, underscoring its robustness across different model architectures and scales. Additional experimental results on more models can be found in Table 5 and 6.

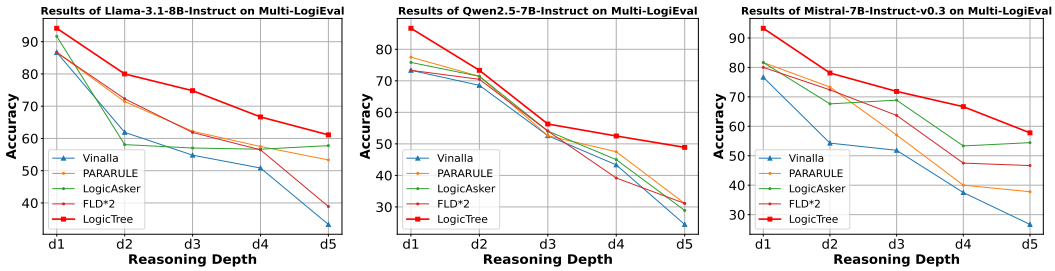

Figure 2: Experiment results for different methods on Multi-LogiEval, which is specifically designed to evaluate the model's multi-step logical reasoning abilities. The horizontal axis represents the number of reasoning steps required to answer the question, ranging from d_1 to d_5. The bold red line at the top represents our method, demonstrating superior performance improvements, especially as the number of reasoning steps increases.

Table 2: Experiment results using Llama3.1-8B across a broader range of task types, covering domains such as logic, math, code, NLI, and others.

Model	Proofwriter	Mathqa	GPQA	Humaneval	Commencenseqa	MNLI	Avg.
Llama-3.1-8B							
vanilla	57.3	42.1	30.8	68.4	73.9	68.1	56.8
+ PARARULE	64.3	42.8	31.9	67.2	73.7	70.5	58.4
+ LogicAsker	69.8	43.2	30.1	68.6	74.5	73.3	59.9
+ FLD$_{\times 2}$	74.1	43.8	31.1	72.3	74.1	75.3	61.8
+ LogicTree	**76.9**	**47.3**	**32.8**	**74.6**	**75.9**	**76.8**	**64.1**

It is worth noting that LogicTree demonstrates superior performance in all baselines, especially in multi-step complex reasoning tasks. While LogicAsker generates a large volume of atomic instruction data that provides some improvement in basic logical reasoning, it remains inferior to our approach. Furthermore, the datasets synthesized by these baselines yield only marginal gains on more challenging reasoning tasks, such as BBH-Logic and AGIEval-AR. In contrast, our method achieves performance gains of up to 13.9%, further demonstrating that **complex reasoning patterns and instantiated reasoning processes play a crucial role in enhancing the model's reasoning capabilities.**

4.3 Results of Multi-step Reasoning

To further evaluate the model's multi-step logical reasoning ability, we assess **LogicTree** and baselines on Multi-LogiEval, which categorizes questions by reasoning steps from one to five. As shown in Figure 2, LogicTree consistently outperforms all baselines across all categories of test data, with particularly strong improvements at higher reasoning depths (d3, d4, d5). These results indicate that LogicTree effectively enhances LLMs' ability to tackle complex multi-step reasoning tasks. In contrast, other methods exhibit a sharp decline in performance as reasoning depth increases, highlighting their limitations in scaling to more complex reasoning scenarios. Their struggle to maintain accuracy on multi-step reasoning tasks suggests that training with simple reasoning data alone is insufficient for significantly improving the model's ability to handle complex reasoning. LogicTree's superior performance on multi-step reasoning tasks underscores the potential benefits of its approach in improving reasoning abilities on multi-step tasks, making it a valuable tool for complex logical reasoning scenarios. Results related to the 70B model can be found in the Appendix B.4.

4.4 Results on Generalization Experiments Across Multiple Domains

To verify the generalization of LogicTree in enhancing model reasoning capabilities, we conducted additional experiments using an 8B model across a broader range of task types, covering domains such as logic, math, code, NLI, and others. As shown in the table, LogicTree enhances the model's reasoning abilities across multiple domains. The improvement is smaller in tasks requiring extensive

Table 3: Ablation results on four types of reasoning tasks, use Llama-3.1-8B as the backbone models. "Avg" means the average accuracy across four benchmarks for each methods. We bold the best results for each benchmark.

Model	LogicBench	LogiQA2.0	BBH	AGIEval		Avg.
				LR	AR	
Llama-3.1-8B						
vanilla	80.0	42.4	39.3	48.4	20.7	46.2
+ LogicTree(num=3)	**90.6**	**53.5**	**53.2**	**56.1**	**26.5**	**55.9**
w/o inst_scenario	86.3	53.5	46.7	54.5	22.6	52.7
w/o inst_reasoning	88.3	52.9	50.1	53.9	24.4	53.9
w/o inst_diversity						
num=1	83.1	51.2	49.6	54.1	23.9	52.4
num=2	88.1	52.9	50.8	52.9	26.1	54.2
num=4	92.6	52.7	53.2	56.1	25.7	55.2
num=5	94.5	50.3	50.8	51.2	23.7	54.1

knowledge recall, such as commonsense and science, since LogicTree primarily strengthens generalization rather than knowledge retention, which is largely dependent on pretraining. However, in reasoning-intensive domains like logic, math, and code, the model exhibits significant gains, demonstrating that LogicTree effectively improves reasoning capabilities through diverse instantiated reasoning processes.

4.5 Ablation Study

In this section, we conduct ablation experiments to assess the contributions of key design components in LogicTree : (1) instantiating real-world reasoning scenarios (**Analysis 1**), (2) incorporating natural language reasoning processes under instantiated scenarios (**Analysis 2**), and (3) ensuring diversity in instantiated scenarios (**Analysis 3**). Specifically, "*w/o inst_scenario*" refers to training on symbolic logical expressions without instantiating the logical reasoning tree into real-world scenarios; "*w/o inst_reasoning*" denotes instantiating real-world scenarios while retaining a fully symbolic representation of the reasoning process; and "*inst_diversity*" means omitting control over the diversity of instantiated scenarios. In particular, we evaluate performance by instantiating one and two reasoning scenarios and processes per logical tree (whereas LogicTree employs three). As shown in Table3, removing any of these components results in a notable performance decline, highlighting their importance in enhancing logical reasoning capabilities.

Analysis 1: Instantiating reasoning scenarios enables the model to learn the ability to apply logical reasoning in specific tasks, rather than simply memorizing language templates or the potential relationships between facts. As shown in Table 3, while symbolic rules enhance reasoning, the improvement without instantiation is significantly less than with LogicTree. This underscores instantiation's importance: it teaches applying logic in complex tasks, not just memorizing rules.

Analysis 2: The reasoning processes in natural language based on instantiated scenarios can further enhance the model's reasoning performance in challenging tasks. As Table 3 demonstrates, instantiating symbolic logical problems into real-world scenarios improved the model's reasoning and accuracy compared to purely symbolic questions, supporting Analysis 1's findings. However, its performance remained inferior to LogicTree. This underscores that natural language reasoning based on instantiated scenarios is crucial for boosting logical ability in complex tasks. On hard problems like AGIEval-AR, this instantiation method yielded only modest gains 1.8%, highlighting the growing value of natural language reasoning on instantiated scenarios for intricate challenges.

Analysis 3: Diverse reasoning scenarios able LLMs to develop more generalizable and transferable logical reasoning abilities. We instantiate different numbers of reasoning scenarios for the same logical reasoning tree, ranging from 1 to 5. When num=1, the model's performance on some

benchmarks is even worse than when no instantiation is used ("w/o instantiation"). This suggests that using only a single reasoning scenario may cause the model to overfit to the relationships between the facts within that scenario, rather than learning genuine reasoning abilities. However, as num increases, the model's performance continues to improve and converges around 3 or 4. Therefore, incorporating more diverse reasoning scenarios proves beneficial for enabling the model to acquire genuinely generalized logical reasoning skills.

5 Related Works

5.1 Synthetic Logic Corpus for Training

With the increasing adoption of synthetic datasets [18, 9, 50], there has been a growing focus on generating high-quality logical reasoning datasets to enhance the reasoning capabilities of LLMs[53]. Early works[7, 39, 3] generated deductive processes by repeatedly applying deduction rules to a given set of facts and transformed these processes into natural language sentences using fixed templates. FLD*2 [30, 29] introduced systematic design principles for logical data synthesis and randomly combined various rules to construct multi-step reasoning datasets. **(author?)** [41] proposed a framework to evaluate the ability of LLMs to handle individual atomic reasoning rules and generate targeted samples for improvement. LogicPro [16] further advanced this line of research by using LLMs to convert algorithmic problems into logical reasoning tasks. However, These methods still have limitations in the complexity of logical trees, producing rules or facts with limited real-world applicability and simplistic language. Our research further extends the generation of logical reasoning trees, encompassing various complex reasoning patterns in first-order logic. Additionally, we use LLMs to instantiate these trees into realistic reasoning scenarios and generate reasoing process in natural language.

5.2 Evaluation of Logic Reasoning

Accurately evaluating the reasoning abilities of LLMs is both essential and challenging. Many studies have focused on assessing this fundamental capability of LLMs[38, 20]. For instance, LogiQA[19] and AR-LAST[55] evaluated models' logical reasoning abilities in real-world scenarios by collecting human examination questions. LogicBench [33] demonstrated that existing LLMs struggle to handle complex logical contexts, even when involving only a single reasoning pattern. **(author?)** [41] introduced LogicAsker, which utilizes a set of atomic reasoning skills to assess the ability of LLMs to learn logical rules. FOLIO[13] and Multi-LogiEval[35] further assessed the multi-step logical reasoning capabilities of LLMs. These studies [8, 4] highlight the significant challenges faced by LLMs in logical reasoning tasks, underscoring the critical need for high-quality training datasets to improve their reasoning performance.

6 Conclusion

In this paper, we propose **LogicTree**, a novel framework for synthesizing instantiated, complex logical reasoning datasets. Our method leverages first-order logic rules to generate complex multi-step logical reasoning trees, which are then instantiated using LLMs to produce natural language reasoning data with realistic scenarios. The resulting synthetic datasets feature intricate reasoning patterns, promoting the development of advanced reasoning abilities in LLMs to address complex tasks. Furthermore, by incorporating instantiated reasoning processes, our method enables LLMs to acquire generalizable reasoning skills, rather than simply memorizing implicit relationships between facts. Extensive experiments across multiple complex logical reasoning benchmarks demonstrate the effectiveness of LogicTree, with an average accuracy improvement of 9.4%. LogicTree consistently outperforms all baselines, particularly in multi-step complex reasoning tasks. Further analysis indicates that both instantiated reasoning processes and the diversity of instantiated scenarios contribute to enhancing the model's generalizable logical reasoning abilities.

Acknowledgments

The authors would like to thank all the anonymous reviewers for their insightful comments and valuable suggestions. This work was supported by the National Key R&D Program of China under contract 2022ZD0119801, and the National Nature Science Foundations of China grants U23A20388, 62021001 and 624B1011.

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[2] Qiming Bao, Alex Yuxuan Peng, Tim Hartill, Neset Tan, Zhenyun Deng, Michael Witbrock, and Jiamou Liu. Multi-step deductive reasoning over natural language: An empirical study on out-of-distribution generalisation. *arXiv preprint arXiv:2207.14000*, 2022.

[3] Qiming Bao, Alex Yuxuan Peng, Tim Hartill, Neset Tan, Zhenyun Deng, Michael Witbrock, and Jiamou Liu. Multi-step deductive reasoning over natural language: An empirical study on out-of-distribution generalisation. *arXiv preprint arXiv:2207.14000*, 2022.

[4] Leonardo Bertolazzi, Albert Gatt, and Raffaella Bernardi. A systematic analysis of large language models as soft reasoners: The case of syllogistic inferences. *arXiv preprint arXiv:2406.11341*, 2024.

[5] Hugo Bronkhorst, Gerrit Roorda, Cor J. M. Suhre, and Martin J. Goedhart. Logical reasoning in formal and everyday reasoning tasks. *International Journal of Science and Mathematics Education*, 18:1673–1694, 2020.

[6] Hanzhu Chen, Xu Shen, Jie Wang, Zehao Wang, Qitan Lv, Junjie He, Rong Wu, Feng Wu, and Jieping Ye. Knowledge graph finetuning enhances knowledge manipulation in large language models. In *The Thirteenth International Conference on Learning Representations*, 2025.

[7] 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*, pages 3882–3890, 2021.

[8] John Dougrez-Lewis, Mahmud Elahi Akhter, Yulan He, and Maria Liakata. Assessing the reasoning abilities of chatgpt in the context of claim verification. *arXiv preprint arXiv:2402.10735*, 2024.

[9] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

[10] Xidong Feng, Ziyu Wan, Muning Wen, Stephen Marcus McAleer, Ying Wen, Weinan Zhang, and Jun Wang. Alphazero-like tree-search can guide large language model decoding and training. *arXiv preprint arXiv:2309.17179*, 2023.

[11] Zijie Geng, Jie Wang, Ziqi Liu, Feng Ju, Yiming Li, Xing Li, Mingxuan Yuan, Jianye Hao, Defu Lian, Enhong Chen, and Feng Wu. Accurate KV cache eviction via anchor direction projection for efficient LLM inference. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

[12] Runquan Gui, Zhihai Wang, Jie Wang, Chi Ma, Huiling Zhen, Mingxuan Yuan, Jianye HAO, Defu Lian, Enhong Chen, and Feng Wu. Hypertree planning: Enhancing llm reasoning via hierarchical thinking. In *Forty-second International Conference on Machine Learning*, 2025.

[13] Simeng Han, Hailey Schoelkopf, Yilun Zhao, Zhenting Qi, Martin Riddell, Wenfei Zhou, James Coady, David Peng, Yujie Qiao, Luke Benson, et al. Folio: Natural language reasoning with first-order logic. *arXiv preprint arXiv:2209.00840*, 2022.

[14] Ruixin Hong, Hongming Zhang, Xinyu Pang, Dong Yu, and Changshui Zhang. A closer look at the self-verification abilities of large language models in logical reasoning. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 900–925, 2024.

[15] Jie Huang and Kevin Chen-Chuan Chang. Towards reasoning in large language models: A survey. *arXiv preprint arXiv:2212.10403*, 2022.

[16] Jin Jiang, Yuchen Yan, Yang Liu, Yonggang Jin, Shuai Peng, Mengdi Zhang, Xunliang Cai, Yixin Cao, Liangcai Gao, and Zhi Tang. Logicpro: Improving complex logical reasoning via program-guided learning. *arXiv preprint arXiv:2409.12929*, 2024.

[17] Xize Liang, Lin Yang, Jie Wang, Yiyang Lu, Runyu Wu, Hanzhu Chen, and Jianye Hao. Boosting multi-domain fine-tuning of large language models through evolving interactions between samples. In *Forty-second International Conference on Machine Learning*, 2025.

[18] Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 2024.

[19] Hanmeng Liu, Jian Liu, Leyang Cui, Zhiyang Teng, Nan Duan, Ming Zhou, and Yue Zhang. Logiqa 2.0—an improved dataset for logical reasoning in natural language understanding. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 2023.

[20] Hanmeng Liu, Ruoxi Ning, Zhiyang Teng, Jian Liu, Qiji Zhou, and Yue Zhang. Evaluating the logical reasoning ability of chatgpt and gpt-4. *arXiv preprint arXiv:2304.03439*, 2023.

[21] Hanmeng Liu, Zhiyang Teng, Leyang Cui, Chaoli Zhang, Qiji Zhou, and Yue Zhang. Logicot: Logical chain-of-thought instruction-tuning. *arXiv preprint arXiv:2305.12147*, 2023.

[22] Haoyang Liu, Jie Wang, Wanbo Zhang, Zijie Geng, Yufei Kuang, Xijun Li, Bin Li, Yongdong Zhang, and Feng Wu. MILP-studio: MILP instance generation via block structure decomposition. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[23] Tongxuan Liu, Wenjiang Xu, Weizhe Huang, Yuting Zeng, Jiaxing Wang, Xingyu Wang, Hailong Yang, and Jing Li. Logic-of-thought: Injecting logic into contexts for full reasoning in large language models. *arXiv preprint arXiv:2409.17539*, 2024.

[24] Man Luo, Shrinidhi Kumbhar, Mihir Parmar, Neeraj Varshney, Pratyay Banerjee, Somak Aditya, Chitta Baral, et al. Towards logiglue: A brief survey and a benchmark for analyzing logical reasoning capabilities of language models. *arXiv preprint arXiv:2310.00836*, 2023.

[25] Qitan Lv, Jie Wang, Hanzhu Chen, Bin Li, Yongdong Zhang, and Feng Wu. Coarse-to-fine highlighting: Reducing knowledge hallucination in large language models. In *International Conference on Machine Learning*, pages 33594–33623. PMLR, 2024.

[26] Qing Lyu, Shreya Havaldar, Adam Stein, Li Zhang, Delip Rao, Eric Wong, Marianna Apidianaki, and Chris Callison-Burch. Faithful chain-of-thought reasoning. In *The 13th International Joint Conference on Natural Language Processing and the 3rd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics (IJCNLP-AACL 2023)*, 2023.

[27] Bill MacCartney and Christopher D Manning. Natural logic for textual inference. In *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*, pages 193–200, 2007.

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

[29] Terufumi Morishita, Gaku Morio, Atsuki Yamaguchi, and Yasuhiro Sogawa. Enhancing reasoning capabilities of llms via principled synthetic logic corpus. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

[30] Terufumi Morishita, Gaku Morio, Atsuki Yamaguchi, and Yasuhiro Sogawa. Learning deductive reasoning from synthetic corpus based on formal logic. In *International Conference on Machine Learning*, pages 25254–25274. PMLR, 2023.

[31] Kentaro Ozeki, Risako Ando, Takanobu Morishita, Hirohiko Abe, Koji Mineshima, and Mitsuhiro Okada. Exploring reasoning biases in large language models through syllogism: Insights from the neubaroco dataset. In *Findings of the 62nd Annual Meeting of the Association for Computational Linguistics, ACL 2024*, pages 16063–16077. Association for Computational Linguistics (ACL), 2024.

[32] Liangming Pan, Alon Albalak, Xinyi Wang, and William Yang Wang. Logic-lm: Empowering large language models with symbolic solvers for faithful logical reasoning. *arXiv preprint arXiv:2305.12295*, 2023.

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

[34] Barbara H. Partee, Alice ter Meulen, and Robert E. Wall. Mathematical methods in linguistics. 1990.

[35] Nisarg Patel, Mohith Kulkarni, Mihir Parmar, Aashna Budhiraja, Mutsumi Nakamura, Neeraj Varshney, and Chitta Baral. Multi-logieval: Towards evaluating multi-step logical reasoning ability of large language models. *arXiv preprint arXiv:2406.17169*, 2024.

[36] Hyun Ryu, Gyeongman Kim, Hyemin S Lee, and Eunho Yang. Divide and translate: Compositional first-order logic translation and verification for complex logical reasoning. *arXiv preprint arXiv:2410.08047*, 2024.

[37] Damien Sileo. Scaling synthetic logical reasoning datasets with context-sensitive declarative grammars. *arXiv preprint arXiv:2406.11035*, 2024.

[38] Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V Le, Ed H Chi, Denny Zhou, et al. Challenging big-bench tasks and whether chain-of-thought can solve them. *arXiv preprint arXiv:2210.09261*, 2022.

[39] Oyvind Tafjord, Bhavana Dalvi Mishra, and Peter Clark. Proofwriter: Generating implications, proofs, and abductive statements over natural language. *arXiv preprint arXiv:2012.13048*, 2020.

[40] Fumiya Uchiyama, Takeshi Kojima, Andrew Gambardella, Qi Cao, Yusuke Iwasawa, and Yutaka Matsuo. Which programming language and what features at pre-training stage affect downstream logical inference performance? In *EMNLP*, 2024.

[41] Yuxuan Wan, Wenxuan Wang, Yiliu Yang, Youliang Yuan, Jen-tse Huang, Pinjia He, Wenxiang Jiao, and Michael Lyu. 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, 2024.

[42] Yuxuan Wan, Wenxuan Wang, Yiliu Yang, Youliang Yuan, Jen-tse Huang, Pinjia He, Wenxiang Jiao, and Michael R Lyu. A & b== b & a: Triggering logical reasoning failures in large language models. *arXiv preprint arXiv:2401.00757*, 2024.

[43] Binwu Wang, Pengkun Wang, Wei Xu, Xu Wang, Yudong Zhang, Kun Wang, and Yang Wang. Kill two birds with one stone: Rethinking data augmentation for deep long-tailed learning. In *The Twelfth International Conference on Learning Representations*, 2024.

[44] Pengkun Wang, Zhe Zhao, HaiBin Wen, Fanfu Wang, Binwu Wang, Qingfu Zhang, and Yang Wang. Llm-autoda: Large language model-driven automatic data augmentation for long-tailed problems. *Advances in Neural Information Processing Systems*, 37:64915–64941, 2024.

[45] Siyuan Wang, Zhongyu Wei, Yejin Choi, and Xiang Ren. Can llms reason with rules? logic scaffolding for stress-testing and improving llms. *arXiv preprint arXiv:2402.11442*, 2024.

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

[47] Chulin Xie, Yangsibo Huang, Chiyuan Zhang, Da Yu, Xinyun Chen, Bill Yuchen Lin, Bo Li, Badih Ghazi, and Ravi Kumar. On memorization of large language models in logical reasoning. *arXiv preprint arXiv:2410.23123*, 2024.

[48] Yuxi Xie, Anirudh Goyal, Wenyue Zheng, Min-Yen Kan, Timothy P Lillicrap, Kenji Kawaguchi, and Michael Shieh. Monte carlo tree search boosts reasoning via iterative preference learning. *arXiv preprint arXiv:2405.00451*, 2024.

[49] Jundong Xu, Hao Fei, Liangming Pan, Qian Liu, Mong-Li Lee, and Wynne Hsu. Faithful logical reasoning via symbolic chain-of-thought. *arXiv preprint arXiv:2405.18357*, 2024.

[50] An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

[51] Tian Ye, Zicheng Xu, Yuanzhi Li, and Zeyuan Allen-Zhu. Physics of language models: Part 2.1, grade-school math and the hidden reasoning process. *arXiv preprint arXiv:2407.20311*, 2024.

[52] Dan Zhang, Sining Zhoubian, Ziniu Hu, Yisong Yue, Yuxiao Dong, and Jie Tang. Rest-mcts*: Llm self-training via process reward guided tree search. *arXiv preprint arXiv:2406.03816*, 2024.

[53] Honghua Zhang, Liunian Harold Li, Tao Meng, Kai-Wei Chang, and Guy Van Den Broeck. On the paradox of learning to reason from data. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence*, pages 3365–3373, 2023.

[54] Yifan Zhang, Yang Yuan, and Andrew Chi-Chih Yao. On the diagram of thought. *arXiv preprint arXiv:2409.10038*, 2024.

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

Impact Statement, Limitations and Future Work

This paper follows existing research on logical reasoning data synthesis and constructs a multi-hop complex logical reasoning dataset for model training, which helps LLMs learn advanced and generalizable logical reasoning capabilities. Furthermore, our synthesized data can be used to precisely evaluate the model's multi-step logical reasoning ability, contributing to research on enhancing models' logical reasoning capabilities. Moreover, our work does not involve human or animal experiments, so the ethical impacts and expected societal implications are those that are well established.

This paper presents work whose goal is to advance the field of logical data synthesis. There are many potential societal consequences of our work, none of which we feel must be specifically highlighted here.

As for the limitations of LogicTree, while it significantly enhances the logical reasoning abilities of LLMs by synthesizing logical datasets, it cannot be integrated with other forms of reasoning. Exploring how to incorporate other reasoning paradigms, such as commonsense reasoning, into logical reasoning data to enhance the comprehensive reasoning abilities of LLMs will be the focus of our future work.

A More Related Works

A.1 Prompt-Based Methods for Enhancing Logical Reasoning

Leveraging large language models (LLMs) for complex logical reasoning problems has been a key focus of recent research. Many efforts[12, 25] focus on designing more effective prompting strategies to help LLMs complete complex logical reasoning tasks. For example, Chain-of-Thought (CoT)[46] guides LLMs to reason step by step nd output the final answer based on the reasoning process. In order to better simulate rational logical thought processes, **(author?)** [54] proposed Diagram-of-Thought (DoT) that models iterative logical reasoning in LLMs as the construction of a directed acyclic graph (DAG). Other approaches involve prompting LLMs to first translate natural language into symbolic language, which is then used to facilitate logical reasoning. Symbolic Chain-of-Thought (SymbCoT) [49] converts input from natural language into a symbolic representation and subsequently constructs a step-by-step reasoning plan based on symbolic logic rules. Logic-of-Thought[23] employs LLMs to extract logical expressions and applies logical reasoning rules to derive extended logical expressions, which are then translated back into natural language to support subsequent reasoning.

Additionally, some research integrates LLMs with external tools, such as solvers, to enhance reasoning capabilities[26, 32]. These approaches first leverage LLMs to convert natural language input into a symbolic form (such as logic programming languages, first-order logic, constraint satisfaction problems, or Boolean satisfiability formulations) compatible with external solvers, which are then used to perform logical reasoning and yield the desired result.

Although prompt-based methods can effectively leverage the potential of large language models (LLMs), their performance remains fundamentally constrained by the models' inherent reasoning capabilities. This limitation is particularly evident in smaller models, which often produce hallucinations and logical errors during inference. On the other hand, approaches that incorporate external solvers primarily exploit the model's ability to translate natural language into symbolic representations, rather than enhancing its intrinsic reasoning ability. In contrast, our approach focuses on synthesizing a large amount of high-quality logical reasoning data from a training perspective, which helps enhance the inherent reasoning capabilities of LLMs.

A.2 Reasoning Data Synthesis

An increasing number of studies have focused on synthesizing complex tasks and high-quality reasoning processes [43, 44, 22], which are utilized for supervised fine-tuning or preference alignment training[6, 17]. For example, the O1 series of studies leverages reinforcement learning and Monte Carlo Tree Search(MCTS) that enable LLMs to optimize reasoning processes autonomously. Benefiting from the extensive tree search space and the guidance of an excellent Process Reward Model, these methods can generate high-quality reasoning processes[52, 10, 48]. However, this approach incurs high computational costs and long search times, while generation efficiency is particularly

crucial in LLM-based settings[11]. In contrast, our method leverages rigorous symbolic logic trees to guide the LLM, offering lower costs and higher correctness.

B More Details about Experiments

B.1 Information of Logical Reasoning Benchmarks

In this study, we selected six representative benchmarks to evaluate the reasoning capabilities of the models. These benchmarks encompass various aspects of logical reasoning, diverse contexts, and difficulty levels, making them suitable for a comprehensive evaluation. As shown in Table 4, we provide brief explanations of each benchmark.

Table 4: Information of Logical Reasoning Benchmarks used in our experiments

Benchmark	Data Count	Explanation
LogicBench	160	LogicBench is a natural language dataset designed to evaluate the logical reasoning abilities of LLMs. In this work, we use the *MCQA* (Multiple Choice Question-Answering) subset of LogicBench(Eval), which focuses on tasks that apply a single inference rule.
LogiQA	1527	LogiQA 2.0 is an enhanced dataset for evaluating logical reasoning in natural language understanding tasks. We use the main part of LogiQA, which includes multiple-choice reading comprehension questions.
FOLIO	203	FOLIO is an expert-crafted dataset for evaluating natural language reasoning with first-order logic. It includes logically complex examples presented in natural language and their formal FOL representations. For our experiment, we use all validation data, leveraging its dual-format structure to precisely assess models' ability to interpret and reason with formal logical constructs.
BBH	1187	BIG-Bench Hard (BBH) is a subset of 23 challenging tasks from the BIG-Bench benchmark, focusing on advanced reasoning. For our experiment, we use three tasks: *Causal Judgment*, evaluating causal reasoning in stories; *Formal Fallacies Syllogisms Negation*, testing logical consistency in argument schemes; and *Logical Deduction*, assessing sequence deduction skills.
AGIEval	740	AGIEval is a benchmark assessing foundation models' abilities through tasks from high-standard exams. For our experiment, we use the *LSAT* (Law School Admission Test) tasks, which evaluate logical reasoning, reading comprehension, and analytical reasoning. These tasks challenge models to analyze complex information and draw accurate conclusions, providing a valuable assessment of their capabilities in legal reasoning and analysis.
Multi-LogiEval	525	Multi-LogiEval is a dataset designed to evaluate LLMs' multi-step reasoning abilities across various logic types. This focus allows for a precise evaluation of LLMs' capabilities in handling logical constructs while maintaining manageable complexity.

B.2 Experiment Setups

This section details our experimental design, covering model selection and the specific training configurations. To rigorously validate the effectiveness of LogicTree, we conduct extensive experiments using a diverse set of prominent open-source models spanning various families and scales.

We utilize models from the Llama-3.1, Mistral-v0.3, Qwen2.5, and Deepseek-R1-Distill families, with parameter scales ranging from 1.5B to 70B. We employ two distinct fine-tuning strategies: full fine-tuning is applied to smaller-scale models (under 8B), while the larger 70B model utilizes LoRA fine-tuning. Llama-3.1-8B, Mistral-7B-v0.3 and Qwen2.5-7B are both trained with a learning rate of $1e-6$. Qwen2.5-1.5B and Qwen2.5-3B are trained with a learning rate of $3e-6$. Llama-3.1-70B, due to its LoRA fine-tuning method, is trained with a higher learning rate of $2e-5$. The training utilizes a maximum context length of 4096 tokens, a global batch size of 128, and is conducted for 3 epochs. DeepSpeed with gradient checkpointing and BF16 precision is used for efficient memory usage. The learning rate scheduler follows a cosine schedule, and the warmup ratio is set to 0.03.

B.3 More results of Different Models

To comprehensively validate the effectiveness of LogicTree, We use advanced reasoning models such as DeepSeek-R1-Distill-Llama-8B and DeepSeek-R1-Distill-Qwen-7B. As shown in the table 5, LogicTree consistently improves the performance of DeepSeek-R1-Distill-Llama-8B and DeepSeek-R1-Distill-Qwen-7B across multiple benchmarks, with average gains of 6.5 % and 4.9 %, respectively. Notably, on logic-intensive benchmarks such as the ML series, the improvements reach up to 13.3 % and 10.9 %. These results further confirm the effectiveness of LogicTree in enhancing the logical reasoning capabilities of LLMs.

Table 5: Results of DeepSeek-R1-Distill series models across multiple logical reasoning benchmarks.

Model	Dataset										Avg
	LogicB.	LogIQA	FOLIO	AGIEval-LR	AGIEval-AR	ML-D1	ML-D2	ML-D3	ML-D4	ML-D5	
DeepSeek-R1-Distill-Qwen-7B											
Vanilla	86.8	52.4	61.2	56.8	32.6	70.0	57.0	60.0	55.0	44.4	57.6
+LogicTree	90.6	53.4	63.4	62.4	34.4	80.9	67.6	62.5	56.6	53.3	62.5
Δ	+3.8	+1.0	+2.2	+5.6	+1.8	+10.9	+10.6	+2.5	+1.6	+9.9	+4.9
DeepSeek-R1-Distill-Llama-8B											
Vanilla	83.1	53.4	53.1	53.3	30.4	78.0	67.1	60.0	49.2	46.6	57.4
+LogicTree	91.2	54.9	57.8	57.3	33.9	86.3	73.3	66.0	62.5	56.6	63.9
Δ	+8.1	+1.5	+4.7	+4.0	+3.5	+7.3	+6.2	+6.0	+13.3	+10.0	+6.5

In addition,we conduct additional experiments using Qwen2.5-1.5B and Qwen2.5-3B to further evaluate the effectiveness of LogicTree on smaller language models. As shown in the Table 6, LogicTree effectively enhances the logical reasoning capabilities of smaller models such as Qwen2.5-1.5B and Qwen2.5-3B, achieving consistent and substantial improvements across multiple benchmarks. Specifically, Qwen2.5-1.5B achieves an average accuracy gain of 7.7 %, and Qwen2.5-3B improves by 9.3 %, demonstrating that our synthesized data remains effective even for smaller-scale models.

Table 6: Results of small models across multiple logical reasoning benchmarks.

Model	Dataset										Avg
	LogicBench	LogiQA2.0	FOLIO	AGIEval-LR	AGIEval-AR	ML-D1	ML-D2	ML-D3	ML-D4	ML-D5	
Qwen2.5-1.5B											
Vanilla	65.0	43.9	48.5	39.5	17.4	65.6	53.3	48.8	43.3	31.1	45.8
+LogicTree	74.4	48.4	53.4	42.4	21.3	80.9	56.2	54.8	54.2	44.4	53.5
Qwen2.5-3B											
Vanilla	74.4	53.9	51.0	53.5	20.9	74.5	55.2	46.5	45.0	36.6	51.2
+LogicTree	83.9	56.2	61.2	55.7	24.4	86.3	67.6	59.3	58.3	50.0	60.5

B.4 More Results on Multi-LogiEval

We have further conducted additional experiments on multi-step reasoning(sec4.3) using LLaMA3.1-70B to validate the robustness of our method. Although the 70B model shows some performance improvement over the 8B model at various reasoning depths, it still faces a decline in accuracy as the number of reasoning steps increases. When the reasoning depth reaches 5, the model's accuracy is only 44%. Our method significantly enhances the model's logical reasoning accuracy across different reasoning depths, demonstrating its superiority. As shown in the table 7, the LLaMA3.1-70B model achieved notable performance, even surpassing the vanilla model and all baseline methods, which highlights the scalability of our approach.

Table 7: Results of Llama-3.1-70B on Multi-LogiEval

Model	Method	d1	d2	d3	d4	d5
Llama-3.1-8B	vanilla	86.67	61.90	54.81	50.83	33.33
	PARARULE	86.67	71.43	62.22	57.50	53.33
	logicAsker	91.66	58.09	57.03	56.66	57.77
	$FLD_{\times 2}$	86.67	72.31	61.85	56.47	38.89
	LogicTree	**94.17**	**80.00**	**74.81**	**66.67**	**61.11**
Mistral-7B-v0.3	vanilla	76.67	54.29	51.85	37.50	26.67
	PARARULE	81.67	73.33	57.04	40.00	37.78
	logicAsker	81.67	67.62	68.89	53.33	54.44
	$FLD_{\times 2}$	80.00	72.38	63.70	47.50	46.67
	LogicTree	**93.30**	**78.10**	**71.85**	**66.67**	**57.78**
Qwen2.5-7B	vanilla	73.33	68.57	52.59	43.33	24.44
	PARARULE	77.50	71.43	52.59	47.50	31.11
	logicAsker	75.83	71.43	54.07	45.00	28.89
	$FLD_{\times 2}$	73.33	70.48	54.07	39.17	31.11
	LogicTree	**86.67**	**73.33**	**56.30**	**52.50**	**48.89**
Llama-3.1-70B	vanilla	90.8	64.7	58.2	61.7	44.4
	PARARULE	93.3	69.5	59.3	63.3	56.7
	LogicAsker	93.3	69.5	60.0	65.8	46.7
	$FLD_{\times 2}$	90.8	66.7	63.9	61.7	46.7
	LogicTree	**96.7**	**73.3**	**76.3**	**74.2**	**60.0**

B.5 Results on Quantitative Evaluation of Synthetic Data

Table 8: Evaluation of logical consistency of synthetic data across multiple models

Model	Consistency
GPT-4o	97.6%
Deepseek-R1	96.8%

Logical Consistency To minimize instantiation errors as much as possible, we designed a two-stage prompting strategy to guide the LLM in instantiating logical reasoning trees. Through this process, the LLM does not need to perform reasoning on its own; it only needs to perform symbolic translation. Additionally, we employed a systematic verification method to filter out errors and filtered out 8.73% of erroneous data, thereby improving the overall data quality. We also conducted additional experiments to evaluate the logical consistency of the filtered instantiated data. Specifically, we prompted multiple LLMs to independently assess whether the logical expressions and their corresponding natural language statements were logically consistent. As shown in Table 8, our data maintained a high level of logical consistency across evaluations by several state-of-the-art models. The prompt used for evaluation with LLMs is as follows:

```
You are an expert in logical reasoning and formal logic. Based on the
correspondence between entities and logical symbols, your task is to
evaluate the consistency between the given natural language statements
and corresponding logical expressions. Please assess whether the logical
expressions accurately translate the logical semantics of the natural
language statements.
```

Realism & Contextual Richness We conducted additional validation experiments to demonstrate the superiority of our synthesized data in terms of realism and contextual richness. Specifically, we employed multiple LLMs to evaluate the synthesized data and compared the results with other methods. The evaluation metrics are defined as follows:

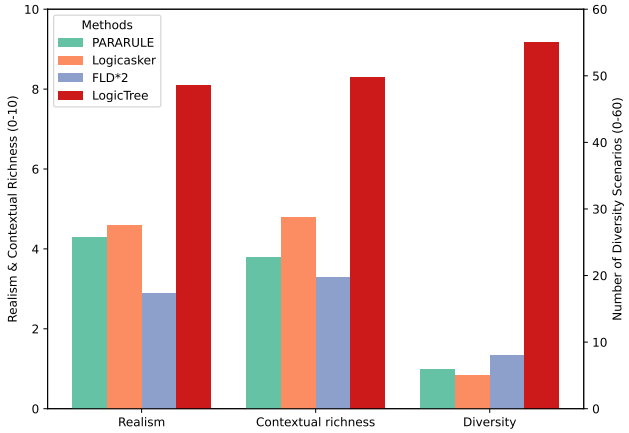

Figure 3: Comparison of Different Methods Across Realism, Contextual Richness, and Diversity. The red bars represent LogicTree, which significantly outperforms other baseline methods.

1. Realism: Assess whether the data presents a realistic, plausible, and logically consistent real-world scenario.

2. Contextual richness: Assess whether the data includes diverse and detailed elements that enrich the scenario.

Notably, due to the rich world knowledge and strong language capabilities of LLMs, LogicTree achieved the highest score for realism and contextual richness in Figure 3. LogicTreeground synthetic reasoning data in concrete and context-rich real-world scenarios that incorporate semantically relevant entities or events, capable of reflecting the complexity and diversity of real-world reasoning tasks. The prompts used for evaluation with LLMs are as follows:

```
You are an expert in data evaluation, contextual analysis, and scenario
validation. Your task is to evaluate the contextual richness of the
generated data, assigning a single score out of 10 to reflect the overall
quality.
The evaluation should consider the following criteria:
1. Contextual Depth: Assess whether the data includes diverse and
detailed elements that enrich the scenario.
2. Variety of Information: Determine whether the generated data presents
a broad range of relevant contextual elements.
Scoring Guidelines:
• 10: Very contextually rich with a wide variety of relevant information.
• 9: Generally rich in context, with sufficient variety and depth, though
slightly lacking in some aspects. • 8: Moderately rich with noticeable
diversity, but certain areas could be expanded to enhance contextual
richness. • 6: Provides basic context with some variety. • 5: Limited
contextual richness, with minimal diversity or detailed elements. • 3:
Poorly developed with very few diverse contextual elements, resulting in a
shallow scenario. • 1: Lacks any meaningful contextual richness, offering
almost no variety or depth.
```

You are an expert in You are an expert in data evaluation, logical reasoning, and real-world scenario validation. Your task is to evaluate the realism of the generated data, assigning a single score out of 10 to reflect the overall quality. The evaluation should consider the following criteria:
1. Realism: Assess whether the data presents a realistic and plausible real-world scenario.
2. Logical Consistency: Determine whether the generated data maintains internal logical coherence.
Scoring Guidelines:
• 10: Highly realistic and plausible. • 9: Mostly realistic and consistent with minor deviations that do not affect overall plausibility. • 8: Generally realistic and coherent with slight inconsistencies or plausible. • 7: Somewhat realistic but with inconsistencies that slightly affect coherence or plausibility. • 6: Reasonably realistic but with noticeable flaws that slightly impact plausibility and background consistency.. • 5: Lacks realism and consistency, with major flaws in logical coherence and background alignment. • 3: Highly unrealistic and implausible, with multiple errors and misalignments. • 1: Completely unrealistic and logically incoherent.

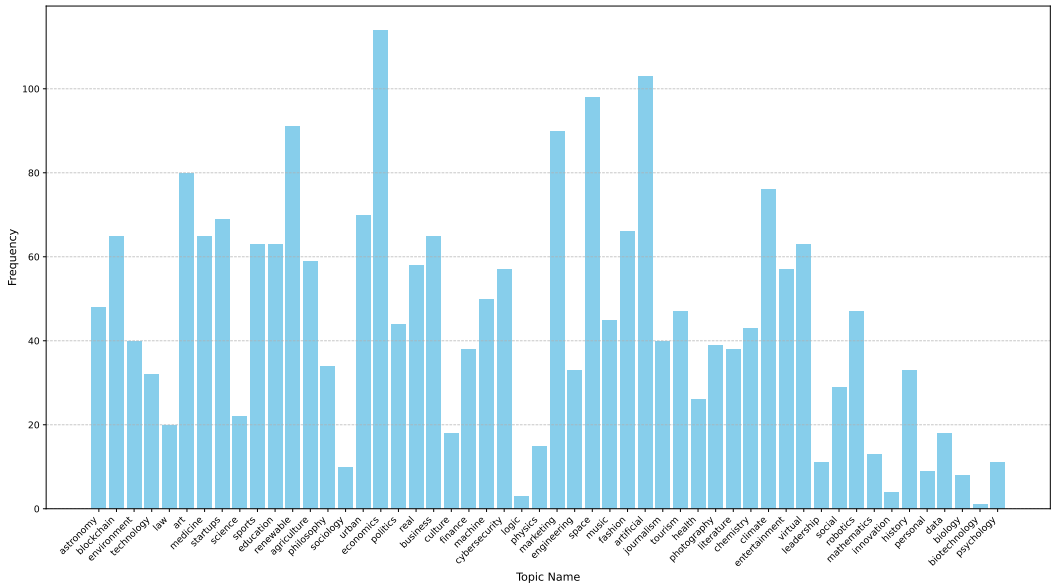

Figure 4: Statistical analysis of the diversity of the synthesized scenarios. We summarized over 50 common themes and prompted the LLM to instantiate multiple real-world scenarios for each logical reasoning tree.

Diversity Scenarios Our goal in introducing instantiation diversity is to enable the model to truly learn logical reasoning rules rather than simply memorizing implicit relationships between specific contents. To achieve this, we summarized over 50 common themes and prompted the LLM to instantiate multiple real-world scenarios for each logical reasoning tree. We also provided a statistical analysis(Figure 4) of the diversity of the synthesized scenarios to demonstrate the effectiveness of this approach.

C More Details about Logical Reasoning Tree Generation

C.1 Logical Reasoning Tree Generation Process

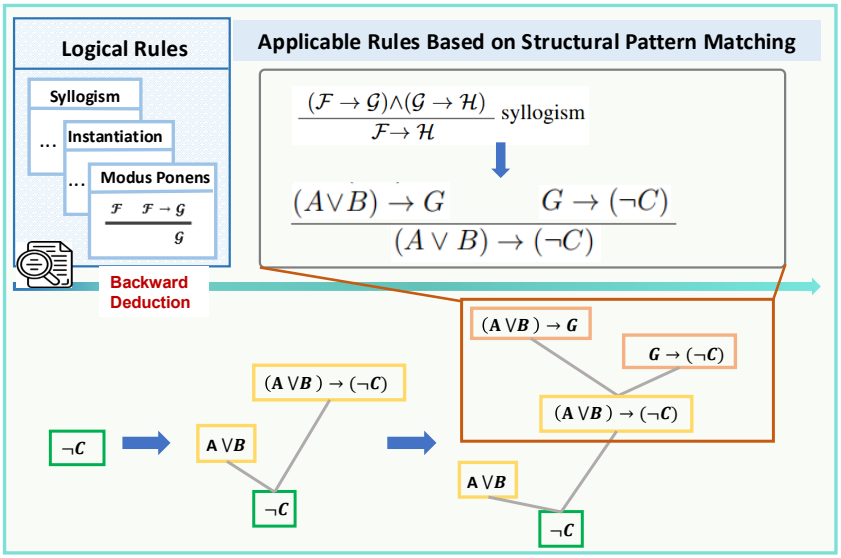

Figure 5: More Details about Logical Reasoning Tree Generation

To clarify our computational cost, we analyzed the number of LLM output tokens consumed when generating each data instance. Specifically, assuming that each reasoning step consumes T output tokens, the computational cost of generating a data instance with reasoning steps m is as follows for each method. We define **Instantiation Cost** and **Verification Cost**.

1. For the **Instantiation Cost**:1) The first call to LLM populates the abstract logical symbols with concrete natural language entities. We denote the number of output tokens consumed in this step as c. 2) The second call translates the sequence of reasoning steps into natural language. We assume that translating each individual reasoning step requires an average of output tokens$T$. The computational cost of generating a data instance with reasoning steps m is:
$$Cost_{inst} = m \cdot T + c$$
2. For the **Verification Cost**:We assume that the complexity of verifying a single step is comparable to that of translating it during instantiation. Consequently, the computational cost for verifying a data instance with m reasoning steps is:
$$Cost_{ver} = m \cdot T$$
.

As shown in the table 9 (the data in the parentheses), the ratio of synthesized tokens to consumed tokens for our method is $\frac{mT}{2mT+c}$ (close to 1/2, in our data $T = 79 \pm 14$, $c = 114 \pm 44$), which is significantly lower than the cost of other methods. Our framework's guarantee of correctness eliminates the need for multiple LLM calls. Consequently, its generation cost is entirely acceptable in comparison with other approaches.

C.2 Statistical Information of Key Properties for logical synthetic data

In order to construct logical reasoning tree, previous methods typically concatenate rules iteratively from a predefined set and randomly replace atomic formulas with complex ones, resulting in an uncontrollable generation process. Moreover, rules are only concatenated when the conclusion of one

Table 9: A comparison of the generation costs for different LLM-based data synthesis methods.

Method	V-STaR	ALPHALLM	LogicTree
Number of LLM invocations	k	mbk	3
Number of consumed LLM output tokens	mkt	$mbk(T_{\text{rollout}})$ + mT	$2mT + c$
Number of synthesized tokens	mT	mT	mT
Explanation	Where k is the number of candidate reasoning trajectories.	Where b is the number of candidate reasoning nodes per step, k is the number of rollout simulations for each node, and T_{rollout} is the additional tokens consumed during the rollout process.	where c denotes the tokens consumed during the first invocation to generate the instantiated entities.

matches the premise of another, which restricts the diversity and complexity of reasoning patterns (e.g., preventing the integration of propositional and first-order logic rules).

In contrast, we designed a backward deduction method based on structural pattern matching of formulas, which does not require the formula to be identical to the conclusion of the logic rule; instead, it only compares whether the structural patterns of their are matched. In this way, we generate multi-step logical reasoning trees that incorporate diverse and complex reasoning patterns, encompassing both propositional logic and first-order logic. In Table 10, we provide additional details about our synthetic data and compare it with other methods. And in table 11, we present supplementary statistical information to further characterize our dataset.

Table 10: The comparison of synthetic logic corpora, which focuses on several key characteristics: the number of logical rules, the reasoning depth, the symbolic-to-natural language translation method, and the instantiation approach.

	Logic Rules	**Reasoning Steps**	**Translation**	**Instantiation**
RuleTaker	2	1-5	Template	Random Entities
PARARULE	2	1-5	Template	Random Entities
FLD	13	1-8	Template	WorldNet
$\text{FLD}_{\times 2}$	≈ 50	1-8	Template	WorldNet
LogicTree	**190**	**1-15**	**LLM-based**	**Realistic Scenario**

Table 11: Statistical information for logic trees with 5 to 8 reasoning steps, including (1) the average maximum depth of the trees, (2) the average number of nodes per tree, (3) the average number of distinct rules applied, as well as (4) the proportion of first-order logic rules and propositional logic rules used in each tree.

Step	**Number of Nodes**	**Maximum Depth**	**Number of Distinct rules**	**Proportion of Fol**	**Proportion of Prop**
5	11.46 ± 1.19	3.68 ± 1.42	4.14 ± 0.52	31.42	68.58
6	13.46 ± 1.01	5.04 ± 0.32	4.75 ± 0.81	30.45	69.55
7	15.78 ± 1.81	5.61 ± 0.56	5.29 ± 0.72	29.94	70.06
8	18.30 ± 2.01	6.04 ± 0.76	5.67 ± 0.97	28.82	71.18

C.3 Logical Rules used in LogicTree

This section presents the logical rules employed in the LogicTree generation process. These rules are fundamental to automated reasoning and inference generation, serving as the foundation for constructing structured logical trees. Drawn from both First-order Logic and Propositional Logic, the applied rules encompass a variety of logical inference patterns, including Modus Ponens, Hypothetical Syllogism, Disjunctive Syllogism, and Universal Instantiation. Table 12 presents the symbolic form and natural language explanation for a selection of common logical rules.

Table 12: Some Logical Rules Used in LogicTree Generation Process and Their Explanations

Rule Name	Rule Symbol	Explanation
First-order Logic		
MP	$(\forall x(p(x) \to q(x)) \land p(a)) \vdash q(a)$	Modus Ponens: Universal elimination combined with conjunction provides the result $q(a)$.
MT	$(\forall x(p(x) \to q(x)) \land \neg q(a)) \vdash \neg p(a)$	Modus Tollens: From universal elimination and negation, derives $\neg p(a)$.
HS	$(\forall x((p(x) \to q(x)) \land (q(x) \to r(x)))) \vdash (p(a) \to r(a))$	Hypothetical Syllogism: Nested implications for quantified variables result in the conditional $p(a) \to r(a)$.
DS	$(\forall x(p(x) \lor q(x)) \land \neg p(a)) \vdash q(a)$	Disjunctive Syllogism: Disjunction with universal quantification simplifies to $q(a)$ if $\neg p(a)$ is given.
UI	$\forall x(p(x)) \vdash p(a)$	Universal Instantiation: From the universal quantifier $\forall x(p(x))$, deduce the specific instance $p(a)$.
Propositional Logic		
MP	$((p \to q) \land p) \vdash q$	Modus Ponens: If $p \to q$ and p, then q.
DS	$((p \lor q) \land \neg p) \vdash q$	Disjunctive Syllogism: If $p \lor q$ and $\neg p$, then q.
MT	$((p \to q) \land \neg q) \vdash \neg p$	Modus Tollens: If $p \to q$ and $\neg q$, then $\neg p$.
HS	$((p \to q) \land (q \to r)) \vdash (p \to r)$	Hypothetical Syllogism: If $p \to q$ and $q \to r$, then $p \to r$.
MI	$(p \to q) \dashv\vdash (\neg p \lor q)$	Material Implication: Expresses conditional as disjunction.
DMT	$\neg(p \land q) \dashv\vdash \neg p \lor \neg q$	De Morgan's Theorem: Simplifies negation of conjunctions.
CD	$((p \to q) \land (r \to s) \land (p \lor r)) \vdash (q \lor s)$	Constructive Dilemma: Combines conditional and disjunctive reasoning.
DD	$((p \to q) \land (r \to s) \land (\neg q \lor \neg s)) \vdash (\neg p \lor \neg r)$	Destructive Dilemma: If one or more results fail, one or more premises fail.
BD	$((p \to q) \land (r \to s) \land (p \lor \neg s)) \vdash (q \lor \neg r)$	Bipolar Dilemma: Variation of dilemma reasoning combining conditions.

D Details about Prompts for Scenario Instantiation

This section discusses the prompts used during the reasoning scenario instantiation step, detailing their structure and role in guiding the model to produce relevant and coherent outputs for logic trees.

D.1 Logical Tree Instantiation

As shown in Figure 6, our logical tree instantiation prompt primarily consists of five components: 1. Role-play: Setting the context for the model to assume a relevant role. 2. Instantiation instruction: Guiding the model to instantiate logical symbols using appropriate real-world scenarios. 3. Logical expression translation instruction: Assisting the model in understanding key logical relationships by providing in-context learning examples for translation. 4. Diversity control: Selecting different instantiation scenarios to ensure diversity. 5. Specific instantiation in-context learning examples: Offering concrete examples to further guide the instantiation process.

You are a highly skilled logic analyst with expertise in understanding and interpreting complex logical relationships.

Please deeply understand the logical rules between the propositions. Note the meanings of the logical symbols:
- '~' (NOT): Negation, indicates the proposition is not true.
- '>' (IMPLIES):Implies, indicates that there is a inferential relationship between the two propositions.
-'|' (OR): Logical disjunction, indicates that at least one of the propositions on either side is true.
- '&' (AND): Logical conjunction, indicates that both propositions on either side are true.
- '∀x' (FOR ALL): Universal quantifier, indicates that a statement is true for all values of x.

While satisfying the logical rules, replace the symbolic propositions with real life complex events from the most relevant field. Please think step by step to ensure that the replaced events can naturally satisfy the logic rules.

Use the corresponding events to explain each logic rule expression in natural language. Please follow the priority order in the logical expression, think step by step, and ensure the translation is accurate and clear.
For example:
i: P1>(~((~P0)&Q))----If P1 is true, either P0 is true, or Q is false, or both.
ii: (~(Q|R0))>P2----If both Q and R0 are false, then P2 must be true.
iii:(~P2)|(P0|Q)----Either P2 is false, or at least one of P0 or Q is true.
iv: ((P1&P2)>((~Q)>P0))----If both P1 and P2 are true, then [if Q is false, P0 must also be true.].
v: (∀x(P4(x)))----For all x, P4(x) is true.

Generate 3 examples from multiple domains [{domain}], We hope you can leverage relevant entities in this field to construct the aforementioned complex events.

----------example1----------{example1}
----------example2----------{example2}
----------example3----------{example3}

Figure 6: Prompt for Logical Tree Instantiation

D.2 Reasoning Process Generation

As shown in Figure 7, our reasoning process generation prompt primarily consists of five components: 1. Role-play: Setting the context for the model to assume a relevant role. 2. Understanding of logical relations: It requires recognizing logical operators, understanding how propositions are structured, and determining how they interact based on formal logic rules. 3. Step-by-Step Reasoning Process Generation: In this stage, a structured sequence of logical steps is generated, following formal inference rules. Each step should be clearly justified to show how the conclusion is logically derived from the premises. 4. Conclusion and Answer Explanation: This part provides the final outcome derived from the reasoning process and offers a concise explanation of why the conclusion is valid. 5. Specific reasoning process generation learning examples: Offering concrete examples to further guide the instantiation process.

You are a highly skilled logic analyst with expertise in understanding and interpreting complex logical relationships.

The following is a logical reasoning context and some conclusions. For each conclusion,the truth value can be Yes, No or Uncertain. And We have extracted the main event entities and logic expressions from the context and conclusions. Note the meanings of the logical symbols:
- '~' (NOT): Negation, indicates the proposition is not true.
- '>' (IMPLIES):Implies, indicates that there is a inferential relationship between the two propositions.
-'|' (OR): Logical disjunction, indicates that at least one of the propositions on either side is true.
- '&' (AND): Logical conjunction, indicates that both propositions on either side are true.
- '∀x' (FOR ALL): Universal quantifier, indicates that a statement is true for all values of x.

Please refer to the following reasoning steps and the extracted entities to provide a detailed reasoning explanation in natural language. For each step of reasoning, we provide the given premises and the conclusions that can be deduced. Please carefully explain each step in detail.

for each conclusion, refer to the reasoning process generated above and provide an explanation that aligns with the corresponding answer.

----------example1----------{example1}
----------example2----------{example2}
----------example3----------{example3}

Figure 7: Prompt for Reasoning Process Generation

E Examples of Synthetic Logical Reasoning data

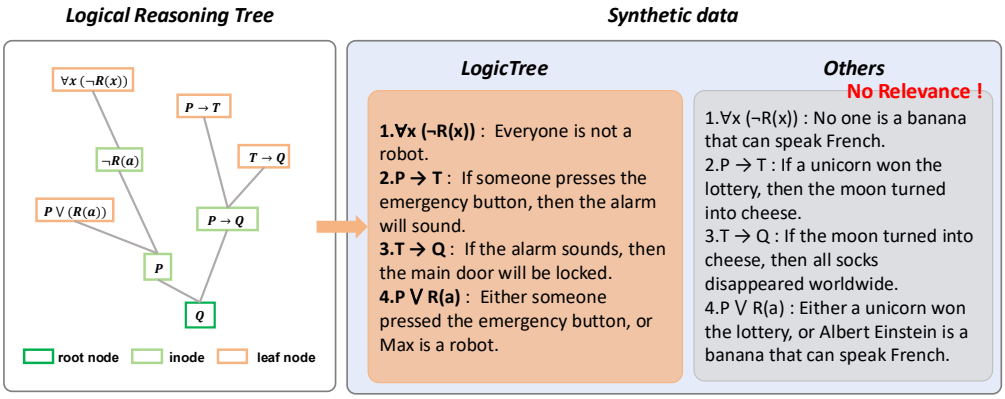

Figure 8: The logical reasoning trees and the corresponding synthetic data are presented, with the left side generated by LogicTree and the right side produced using a template-based random substitution method. It can be observed that the data synthesized by LogicTree maintains a coherent contextual semantic relationship, whereas the data generated by other methods is either unrelated or even contradictory.

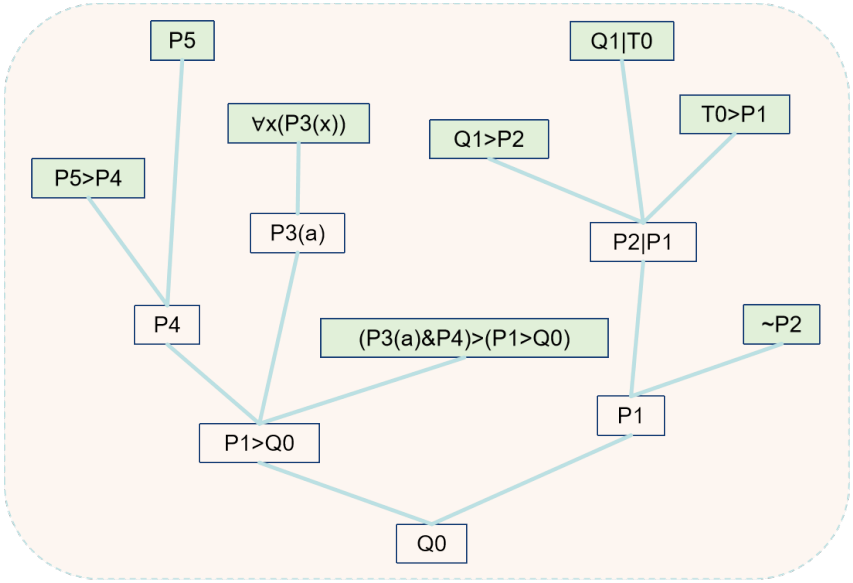

Figure 9: An example of a more complex logical reasoning tree, where green blocks implying the leaves to generate scenarioes

Entities:
•**P1**: Emergency shelters were opened in time.
•**P2**: Casualties occurred.
•**P3(x)**: x is located in a high-risk seismic zone.
•**P4**: Local authorities activated the emergency response protocol.
•**P5**: A magnitude 6.0 or greater earthquake was detected.
•**Q0**: People were safely evacuated.
•**Q1**: Warning sirens were activated.
•**T0**: The national emergency broadcast was triggered.
•**a**: City A

Rules:
rule1: ((P3(a) ∧ P4) → (P1 → Q0)) If City A is in a high-risk seismic zone and local authorities activated the emergency response protocol, then if emergency shelters were opened in time, people were safely evacuated.
rule2: ∀x(P3(x)) All regions are located in high-risk seismic zones.
rule3: (P5 → P4) If a magnitude 6.0 or greater earthquake was detected, then the emergency response protocol was activated.
rule5: (Q1 → P2) If warning sirens were activated, then casualties occurred.
rule6: (T0 → P1) If the national emergency broadcast was triggered, then emergency shelters were opened in time.
rule7: (Q1 ∨ T0) Either warning sirens were activated, or the national emergency broadcast was triggered.
rule8: ¬P2 No casualties occurred.

Reasoning process:

Step1: From rule5 (Q1 → P2) [if warning sirens were activated, then casualties occurred], rule7 (Q1 ∨ T0) [either warning sirens were activated or the national emergency broadcast was triggered], and rule6 (T0 → P1) [if the national emergency broadcast was triggered, then emergency shelters were opened in time], we can deduce that (P2 ∨ P1) [either casualties occurred or emergency shelters were opened in time].
Step2: According to rule2 (∀x(P3(x))) [all cities are located in high-risk seismic zones], we can conclude that (P3(a)) [City A is located in a high-risk seismic zone].
Step3: From rule3 (P5 → P4) [if a magnitude 6.0 or greater earthquake was detected, then the emergency response protocol was activated] and assuming (P5) [a magnitude 6.0+ earthquake was detected], we can conclude that (P4) [the emergency response protocol was activated].
Step4: According to rule1 ((P3(a) ∧ P4) → (P1 → Q0)) [if City A is in a high-risk seismic zone and the emergency response protocol was activated, then if emergency shelters were opened in time, people were safely evacuated], and from step2 (P3(a)) and step3 (P4), we can deduce that (P1 → Q0) [if emergency shelters were opened in time, people were safely evacuated].
Step5: From step1 (P2 ∨ P1) [either casualties occurred or emergency shelters were opened in time] and rule8 (¬P2) [no casualties occurred], we can conclude that (P1) [emergency shelters were opened in time].
Step6: From step4 (P1 → Q0) [if emergency shelters were opened in time, people were safely evacuated] and step5 (P1) [emergency shelters were opened in time], we can conclude that (Q0) [people were safely evacuated].

Figure 10: The Generated Entities, Rules and Reasoning Process of an Example

conclusion1: people were safely evacuated.
answer: Yes
explanation: From step6, we deduced that people were safely evacuated (Q0). Therefore, the conclusion that people were safely evacuated is true.

conclusion2: emergency shelters were not opened in time
answer: No
explanation: From step5, we deduced that emergency shelters were opened in time (P1). Therefore, the conclusion that emergency shelters were not opened in time is false.

conclusion3: either casualties occurred or emergency shelters were opened in time
answer: No
explanation: From step1, we deduced that either casualties occurred or emergency shelters were opened in time (P2 ∨ P1). From step5, we know emergency shelters were opened in time (P1), making the statement “It is not the case that either casualties occurred or emergency shelters were opened in time” false.

conclusion4: it is not the case that if emergency shelters were opened in time, people were safely evacuated
answer: No
explanation: From step4, we deduced that if emergency shelters were opened in time, people were safely evacuated (P1 → Q0). Therefore, the conclusion that it is not the case that if emergency shelters were opened in time, people were safely evacuated is false.

Figure 11: The Generated Answers of an Example

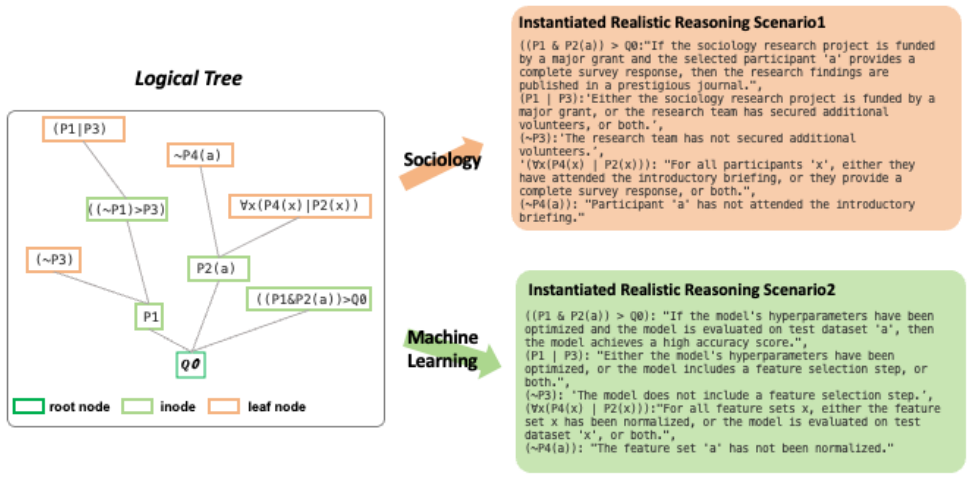

Figure 12: An example of instantiating a logic tree for diverse scenarios

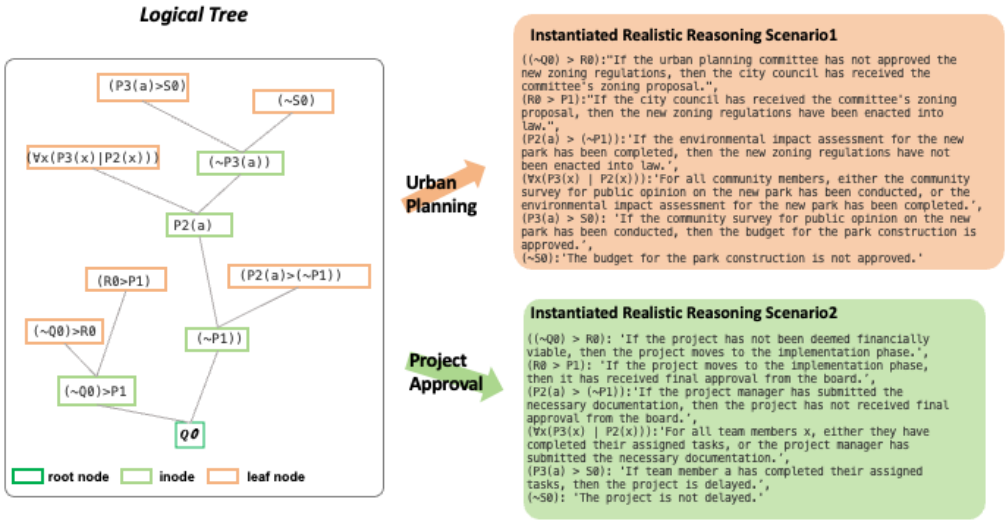

Figure 13: An example of instantiating a logic tree for diverse scenarios

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: In the introduction, we state our main contributions and provide extensive experiments to support these claims.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: In the final section of the article, we discuss the limitations of this work and future directions.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [NA]

Justification: This work primarily focuses on synthetic data and model training.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We provide a detailed explanation of the methodology in the Section 3 and offer a comprehensive experimental setup in the Appendix B.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: we use open-source datasets, and the code used in this paper will be made publicly available upon acceptance.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We offer a comprehensive experimental setup in the Appendix B.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We follow relevant works in this field to report experiment statistical significance

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We offer a comprehensive experimental setup in the Appendix B.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: Our research fully complies with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We discuss this part in detail in the Appendix.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: the paper poses no such risks.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We cite all of the original paper that produced the code package or dataset.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: The paper does not release new assets.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: the paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: the paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [Yes]

Justification: In the method section, we detail the use of LLMs and provide specific prompts.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.