SATQuest: A Verifier for Logical Reasoning Evaluation and Reinforcement Fine-Tuning of LLMs

Yanxiao Zhao[c u l *] **Yaqian Li**[l] **Zihao Bo**[l] **Rinyoichi Takezoe**[l]
Haojia Hui[l] **Mo Guang**[l] **Lei Ren**[l] **Xiaolin Qin**[c u †] **Kaiwen Long**[l †]
[c] Chengdu Institute of Computer Applications, Chinese Academy of Sciences
[u] School of Computer Science and Technology, University of Chinese Academy of Sciences
[l] Li Auto

Abstract

Recent advances in Large Language Models (LLMs) have demonstrated remarkable general reasoning capabilities. However, systematically evaluating and enhancing these reasoning capabilities is challenging due to the lack of controllable and scalable tools for fine-grained analysis. Existing benchmarks and datasets often lack the necessary variable control for multi-dimensional, systematic analysis and training, or have narrow problem types and formats. To address these limitations, we introduce `SATQuest`, a systematic verifier designed to evaluate and enhance logical reasoning in LLMs by generating diverse, Satisfiability-based logical reasoning problems directly from Conjunctive Normal Form (CNF) instances. `SATQuest` structures these problems along three orthogonal dimensions: instance scale, problem type, and question format, employing randomized, SAT-based problem generation and objective answer verification via PySAT. This design mitigates memorization issues, allows for nuanced insights into reasoning performance, and enables effective reinforcement fine-tuning. Our extensive evaluation of various LLMs using `SATQuest` identified significant limitations in their logical reasoning, particularly in generalizing beyond familiar mathematical formats. Furthermore, we show that reinforcement fine-tuning with `SATQuest` rewards substantially improves targeted task performance and generalizes to more complex instances, while highlighting remaining challenges in cross-format adaptation. Through these demonstrations, we showcase `SATQuest`'s potential as a foundational tool and a valuable starting point for advancing LLM logical reasoning.

https://github.com/sdpkjc/SATQuest

1 Introduction

Large Language Models (LLMs) have demonstrated remarkable proficiency in general reasoning tasks, including complex problem-solving and code generation, with models like `o3-mini`[OpenAI, 2025], `DeepSeek-R1`[DeepSeek-AI et al., 2025], and `QwQ-32B`[Team, 2024] excelling in programming, mathematics, and scientific question-answering[Brown et al., 2020, OpenAI et al., 2024, Wei et al., 2022]. This advanced reasoning capability, a cornerstone for Artificial General Intelligence (AGI), is a critical indicator of models' deep understanding and generalization.

To further explore the reasoning capabilities of LLMs, we urgently need evaluation and training tools that are both controllable and scalable. Fine-grained and reliable performance analysis requires

[*]Work done during internship at *Li Auto*.
[†]Corresponding author: `qinxl2001@126.com`, `longkaiwen@lixiang.com`

Preprint. Under review.

systematic variable control, which is fundamental to empirical scientific research. However, existing benchmarks and datasets have significant limitations in variable control, making it difficult to support multi-dimensional, systematic analysis and training experiments—hindering deeper understanding of reasoning mechanisms in LLMs.

Although benchmarks such as *ZebraLogic* [Lin et al., 2025] and *Knights and Knaves* [Xie et al., 2024] have introduced controllable difficulty dimensions, their question types and formats remain narrow, supporting only the *instance scale* dimension. *MATH* [Hendrycks et al., 2021a] and *LiveBench* [White et al., 2025] provide rich content across mathematics and programming; however, they lack structured relationships and rely heavily on human judgment for difficulty labeling. In comparison, general-purpose benchmarks like *GPQA* [Rein et al., 2024], *MMLU* [Hendrycks et al., 2021b], and *Big-Bench* [Srivastava et al., 2023] offer broad coverage, yet suffer from issues such as data leakage, lack of continuity, and insufficient support for multi-dimensional controlled analysis.

In this work, we aim to construct a multi-dimensionally controllable and scalable verifier to support the evaluation and training of LLM reasoning capabilities, enabling the tracking of progress and discovery of limitations in LLM reasoning.

Contributions

- (Sec. 2) `SATQuest` Verifier: To address these limitations, we introduce `SATQuest`, a systematic verifier for evaluating and enhancing logical reasoning in LLMs. `SATQuest` automatically generates diverse logical reasoning problems derived from CNF (Conjunctive Normal Form) instances, structured along three orthogonal dimensions: *instance scale*, *problem type*, and *question format*. This mitigates memorization issues by employing randomized, SAT-based problem generation. Additionally, `SATQuest` incorporates objective and efficient answer verification via PySAT [Ignatiev et al., 2018], enabling effective reinforcement fine-tuning.
- (Sec. 3) Evaluation and Analysis: Using `SATQuest`, we comprehensively evaluated various LLMs (open/closed-weight, vanilla/reasoning) on a reproducible CNF dataset. Our multi-dimensional analysis across *instance scale*, *problem type*, and *question format* revealed significant limitations in current LLMs' logical reasoning abilities, particularly highlighting challenges in generalizing beyond familiar mathematical formats. This evaluation demonstrates `SATQuest`'s effectiveness for generating nuanced insights into reasoning performance.
- (Sec. 4) Verifier-Driven Reinforcement Fine-Tuning: We implemented a Reinforcement Learning from Verifiable Rewards (RLVR) [Lambert et al., 2025] framework utilizing the `SATQuest` verifier. Our results show that fine-tuning LLMs directly using reward signals from `SATQuest` not only improves performance on targeted tasks but also stimulates longer reasoning chains, particularly in structured formats like mathematical notation. We further investigated the generalization of these improvements across varying *instance scales*, *problem types*, and *question formats*, highlighting the potential of verifier-driven RFT to systematically address identified reasoning limitations.

2 SATQuest Challenge Design

Overview. `SATQuest` is a systematic verifier engineered to comprehensively evaluate and enhance the logical reasoning capabilities of LLMs. Its primary goal is to offer a framework for fine-grained analysis, providing deeper insights into the strengths and limitations of LLMs in logical deduction. `SATQuest` is not designed to train LLMs as general-purpose solvers or to make them surpass specialized symbolic solvers in speed or accuracy.

To achieve its objectives, `SATQuest` automatically generates a variety of logical reasoning tasks directly derived from CNF instances. These tasks are meticulously organized along three orthogonal dimensions: *instance scale and difficulty*, *problem type*, and *question format*, each targeting distinct aspects of logical reasoning. This multi-dimensional structure creates a comprehensive and controllable challenge space suitable for nuanced evaluation and effective reinforcement fine-tuning. CNF instances are utilized as the foundational elements due to their formal clarity, their established role as a standard representation in propositional logic, and their inherent compatibility with established SAT solvers for objective answer verification, making them an ideal medium for systematically probing the logical capabilities of LLMs.

Data Generation. `SATQuest` supports evaluation using any CNF instance dataset stored in the standard `DIMACS` format. For reproducibility, we generated two CNF datasets using the procedure outlined in Algorithm 1:

- 🤗 `sdpkjc/SATQuest`: This dataset is generated for evaluation purposes (Sec. 3), specifically to assess logical reasoning across varying instance scales and difficulties. It consists of randomly generated CNF instances with $n \in [3, 16]$ variables and a fixed clause-to-variable ratio resulting in $m = 4n$ clauses. For each (n, m) configuration, 10 CNF instance pairs (one satisfiable and one unsatisfiable) were generated, resulting in a total of 140 CNF pairs. The two CNF instances share the same number of literals and nearly identical CNF structures.
- 🤗 `sdpkjc/SATQuest-RFT-3k`: This dataset is generated for reinforcement fine-tuning (Sec. 4). It consists of CNF instances with $n \in [3, 8]$ variables and clause counts m determined by varying the clause-to-variable ratio from 2.1 to 4.0 in increments of 0.1 (i.e., m ranges from $2.1n$ to $4.0n$). For each (n, m) configuration, 25 CNF instance pairs were generated, resulting in a total of $3,000$ CNF pairs.

Challenge Dimensions 1: *Instance Scale and Difficulty*. We categorize instances by their *scale* (number of variables n, clauses m, and literals) and inherent *difficulty*. Computational *difficulty* is assessed using SAT solver statistics: *decisions*, indicating search breadth; *conflicts*, reflecting constraint-driven backtracking; and *propagations*, quantifying chained logical implications. These structural and solver-derived metrics provide a multi-faceted characterization of an instance's combinatorial complexity, where higher values generally correspond to more challenging problems.

Challenge Dimensions 2: *Problem Type*. We define five fundamental SAT-based problems over a CNF formula $F = \bigwedge_{i=1}^{m} C_i$ on variables $X = \{x_1, \ldots, x_n\}$:

- `SATDP` (SAT Decision Problem): Determine whether F is satisfiable:
$$\texttt{SATDP}(F) = \begin{cases} 1, & \exists\, \alpha : X \to \{0,1\} \text{ such that } F(\alpha) = 1, \\ 0, & \text{otherwise.} \end{cases}$$
Tests the fundamental ability to determine logical consistency.
- `SATSP` (SAT Search Problem): If F is satisfiable, find an assignment α:
$$\alpha \text{ s.t. } F(\alpha) = 1.$$
Probes constructive reasoning by requiring the generation of a satisfying assignment.
- `MaxSAT` (Maximum Satisfiability): Find the assignment α^* that maximizes the number of satisfied clauses:
$$\alpha^* = \arg\max_{\alpha} \sum_{i=1}^{m} \mathbf{1}[C_i(\alpha) = 1].$$
Evaluates optimization skills when maximizing clause satisfaction under conflicting constraints.
- `MCS` (Minimal Correction Subset): For an unsatisfiable F, find a minimal set S whose removal yields a satisfiable formula:
$$S \subseteq \{1, \ldots, m\} \text{ s.t. } \bigwedge_{i \notin S} C_i \text{ is satisfiable and } \forall S' \subset S, \bigwedge_{i \notin S'} C_i \text{ is unsatisfiable}$$
Tests diagnostic reasoning through the identification of minimal corrections for unsatisfiability.
- `MUS` (Minimal Unsatisfiable Subset): For an unsatisfiable F, find a minimal unsatisfiable core S:
$$S \subseteq \{1, \ldots, m\} \text{ s.t. } \bigwedge_{i \in S} C_i \text{ is unsatisfiable and } \forall S' \subset S, \bigwedge_{i \in S'} C_i \text{ is satisfiable}$$
Probes diagnostic reasoning by localizing minimal sources of logical inconsistency.

Each CNF instance pair consists of one satisfiable and one unsatisfiable formula. The satisfiable instance is used for `SATSP`, while the unsatisfiable instance is used for `MaxSAT`, `MCS`, and `MUS`. For `SATDP`, both instances are evaluated, forming two sub-tasks (`SATDP-sat` and `SATDP-unsat`). A response to `SATDP` is considered correct only if both sub-tasks are answered correctly, thereby discouraging random guessing. These problems progressively challenge LLM reasoning capabilities, from foundational deduction and solution construction (`SATDP`/`SATSP`), to constrained optimization (`MaxSAT`), and finally to minimal cause identification and correction (`MCS`/`MUS`).

Challenge Dimensions 3: *Question Format*. Recognizing that the presentation of a problem can significantly impact an LLM's reasoning process, this dimension introduces four logically equivalent representational formats for each CNF instance. This variation aims to test different reasoning skills and reduce reliance on superficial pattern matching.

- `Math` (mathematical notation): Uses $\wedge$, $\vee$, and $\neg$ to represent logic formulas. Balances between formality and readability.
  Example: $x_1 \vee \neg x_2 \vee x_3$
- `DIMACS` (machine format): A minimal, line-based format for representing Boolean formulas in CNF, the standard input for many SAT symbolic solvers.
  Example:
  ```
  p cnf 3 1
  1 -2 3 0
  ```
- `Story` (OR semantics, cookie day scenario): Wraps clauses as friendly narratives—"Alice is happy if..."—to test LLMs' ability to ground disjunctions in natural language.
  Example: "Alice will be happy if she gets crunchy choco (x_1), chewy vanilla ($\neg x_2$), or crunchy peanut (x_3)."
- `DualStory` (AND semantics, cookie day scenario): Presents the negated form—"Alice will be unhappy only if..." —turning OR into AND and requiring semantic tracking.
  Example: "Alice will be unhappy only if she is served crunchy choco (x_1), chewy vanilla ($\neg x_2$), and crunchy peanut (x_3)."

`Math` is common in training data and accessible to math-tuned LLMs, whereas `DIMACS` is a compact, noise-free, machine-readable format that tests a model's ability to interpret raw clause structures. It is specifically designed for evaluating LLMs without relying on mathematical training. `Story` and `DualStory` introduce narrative elements that add informational noise and require the model to translate natural-language logical structures into formal logic before reasoning.

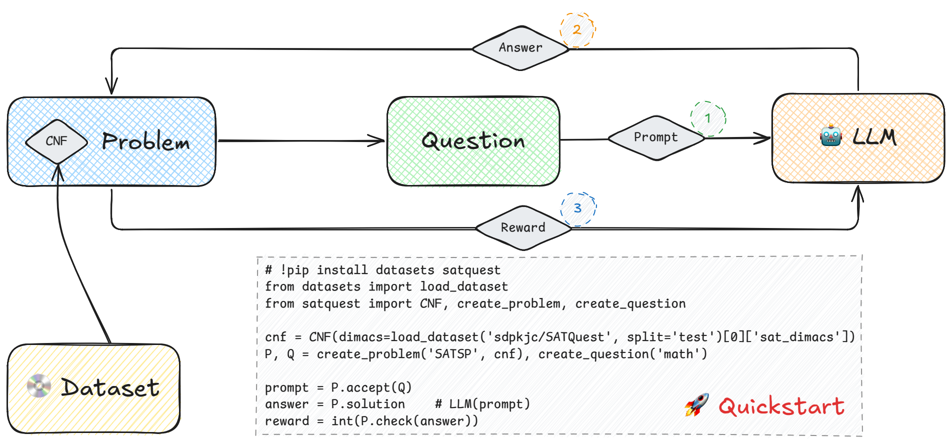

Figure 1: `SATQuest` Pipeline and Quickstart.

Answer Verification. Prior logical reasoning benchmarks (e.g., [Lin et al., 2025, Xie et al., 2024, Mondorf and Plank, 2024]) typically require LLMs to produce `JSON` outputs, choose from preset options, or use complex extraction—methods that hinder automation and scalability despite reduced manual overhead. Moreover, multiple-choice formats can significantly alter the true difficulty of problems, as the complexity of `SATSP` differs fundamentally from SAT verification. `SATQuest` simplifies evaluation by instructing LLMs to output binary strings—1 bit for `SATDP`, n bits for `SATSP`/`MaxSAT`, and m bits for `MCS`/`MUS`. These strings are extracted using regex and checked against CNF constraints via PySAT, allowing for multiple valid answers. A known limitation is that long binary outputs may challenge smaller models' ability to adhere to the expected format. To shed light on this, Fig. 7 presents format correctness statistics for mainstream models evaluated. For the complete prompt and detailed output format instructions, refer to App. B. The `SATQuest` Pipeline and Quickstart are illustrated in Fig. 1.

3 Evaluation

Overview. We conduct a comprehensive evaluation of the logical reasoning performance of various LLMs using the `SATQuest` benchmark. The analysis spans different *instance scales*, *problem types*, and *question formats*, enabling fine-grained, multi-dimensional insights. We begin with the overall benchmark results, followed by detailed analyses along each dimension.

Setup. We evaluate a diverse set of state-of-the-art open-weight and closed-weight LLMs, including vanilla models (`GPT-4.1`, `Qwen2.5-7B/32B-Instruct`, `DeepSeek-V3-0324`), reasoning models (`o3-mini`, `DeepSeek-R1`, `QwQ-32B`), and distilled variants (`DeepSeek-R1-Distill-Qwen-7B/32B`). The evaluation uses the 🤗 `sdpkjc/SATQuest` dataset, comprising 140 CNF instance pairs categorized by scale ($n \in [3, 16]$, clauses $m = 4n$). Each CNF pair yields tasks across five logical reasoning types (`SATDP`, `SATSP`, `MaxSAT`, `MCS`, `MUS`) and four question formats (`Math`, `DIMACS`, `Story`, `DualStory`), resulting in 20 evaluations per CNF pair. For detailed evaluation configurations and parameters, see App. D.

Overall Results. Fig. 2 shows model accuracies on `SATQuest`. `o3-mini` leads with 0.56 accuracy, followed by `DeepSeek-R1` (0.42), `QwQ-32B` (0.40), and `DeepSeek-R1-Distill-Qwen-32B` (0.39), indicating that reasoning-enhanced models outperform vanilla LLMs. Large vanilla models like `GPT-4.1` (0.38) and `DeepSeek-V3-0324` (0.36) perform competitively despite lacking explicit reasoning training. In contrast, smaller vanilla models (e.g., `Qwen2.5-7B-Instruct`) achieve below 0.1 accuracy, revealing limited reasoning capabilities.

These results point to two notable trends. First, reasoning models consistently outperform vanilla counterparts, particularly on more complex tasks. Second, the modest overall accuracy across models reflects the challenging nature of `SATQuest` and its effectiveness in distinguishing reasoning capabilities. Moreover, we observe that performance on `SATQuest` is highly correlated with other recent reasoning benchmarks, such as *GPQA* [Rein et al., 2024] and *ZebraLogic* [Lin et al., 2025], suggesting that `SATQuest` captures essential generalization and reasoning capabilities in LLMs.

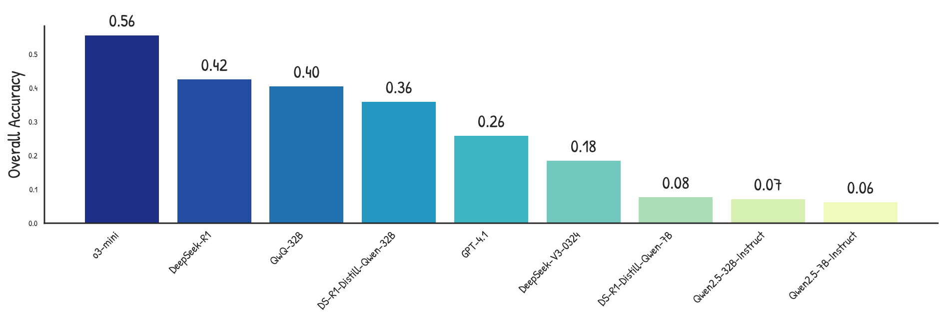

Figure 2: Overall accuracy of evaluated LLMs on the `SATQuest` benchmark, averaged across all *problem types* and *question formats*.

Analysis by *Instance Scale and Difficulty*. We examine how model performance scales with instance complexity, measured using the number of decisions made by established SAT solvers (Glucose 4.1 [Audemard and Simon, 2018] for `SATDP`/`SATSP`, RC2 [Ignatiev et al., 2019] for `MaxSAT`, LBX [Mencía et al., 2015] for `MCS`, and MUSX [Marques-Silva, 2010] for `MUS`). Fig. 3 visualizes model accuracy and response length against this complexity metric. A consistent trend across tasks is that as instance complexity increases (more solver decisions), model accuracy tends to decline, while response length generally increases. Notably, we observe a concerning hallucination phenomenon when models encounter highly complex instances: they often fabricate solver calls or invent simplified reasoning paths rather than engaging with the full logical complexity of the problem. This hallucination is particularly evident for `o3-mini` on `MCS-Math` and `MUS-Math` tasks, where response length actually decreases at high complexity, indicating the model abandons complete reasoning in favor of hallucinated shortcuts. Overall, all models exhibit reduced accuracy on larger and more difficult instances. Top-performing models like `o3-mini` and `DeepSeek-R1` show more gradual degradation, indicating better scalability, whereas less capable models experience a sharp performance

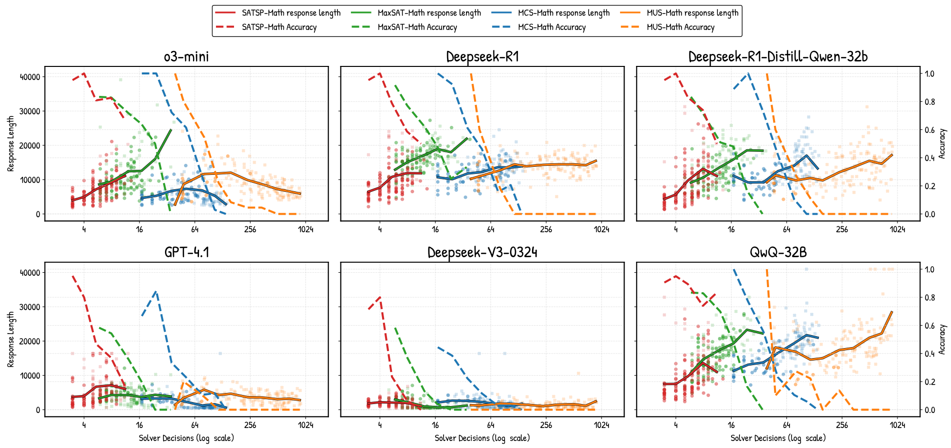

Figure 3: Accuracy (solid lines, right axis) and response length (dashed lines, left axis) vs. instance complexity (solver decisions) for SATSP-Math, MaxSAT-Math, MCS-Math and MUS-Math tasks.

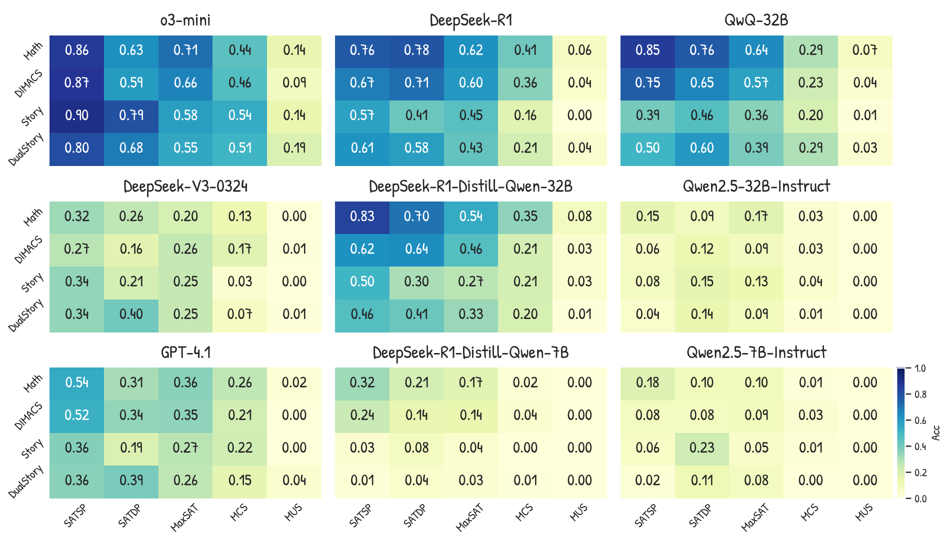

Figure 4: Accuracy heatmaps showing LLM performance breakdown by *problem type* (columns) and *question format* (rows).

drop, often failing completely on moderately complex instances and resorting to increasingly severe hallucinations when faced with logical complexity beyond their reasoning capacity.

Analysis by *Problem Type*. We analyze performance variations across the five distinct SAT-based problem types: SATDP, SATSP, MaxSAT, MCS, and MUS. The heatmaps in Fig. 4 reveal performance differences across these tasks. Consistently across models, performance tends to be highest on SATDP (especially SATDP-sat) and SATSP, and lowest on MCS and MUS, forming a clear difficulty hierarchy. Models generally handle the basic decision (SATDP) and search (SATSP) tasks better than tasks requiring optimization or diagnosis, though performance on SATSP is often lower than SATDP. Performance on the optimization task MaxSAT typically sits between the basic tasks and the diagnostic tasks. The diagnostic tasks, MCS and MUS, which require identifying minimal subsets, prove particularly challenging, with accuracy dropping significantly for almost all models. While top models like o3-mini maintain some capability even on harder tasks, the gap between task types is pronounced across the board. This performance stratification aligns with the solver complexity shown in Fig. 3 (measured by solver decisions), suggesting LLMs struggle progressively more with tasks demanding global optimization, minimality constraints, and diagnostic reasoning over combinatorial spaces. Overall, the results highlight LLM limitations in tackling the full spectrum of logical reasoning challenges represented by these diverse SAT-based tasks.

Analysis by *Question Format*. The way a logical problem is presented can significantly affect an LLM's ability to solve it. We analyze this impact by evaluating performance across four distinct question formats: `Math`, `DIMACS`, `Story`, and `DualStory`. Fig. 4 illustrates how accuracy varies across these formats. All models perform best in the `Math` format, generally achieving their highest accuracy, followed by `DIMACS`, with `Story` and `DualStory` formats yielding the lowest accuracy. `o3-mini` demonstrates relatively stable performance across the four formats, indicating strong reasoning robustness regardless of presentation style. However, other open-weight reasoning models like `DeepSeek-R1` and `QwQ-32B`, while performing well in the `Math` format, exhibit a significant drop in accuracy in other formats, suggesting higher sensitivity to the presentation style.

It is noteworthy that the `Story` and `DualStory` problems introduce narrative elements, adding informational noise and requiring the model to translate the natural language logical structure into formal logic before reasoning. The increased difficulty and subsequent lower accuracy are thus expected. However, the `DIMACS` format is structurally similar to `Math`, contains no redundant information, and has higher information density. Despite this, open-weight reasoning models still show a marked decrease in accuracy compared to `Math` (e.g., `DeepSeek-R1` and `QwQ-32B` accuracy dropped by 9% and 10% respectively on `SATSP`-`DIMACS` compared to `SATSP`-`Math`).

Through case studies presented in App. C, we observe that `DeepSeek-R1` and `QwQ-32B` often attempt to reason directly within the `DIMACS` format rather than translating it into formal mathematical notation. This approach involves working with the raw `DIMACS` clauses, which requires tracking multiple variable assignments simultaneously across numerous constraints. The models frequently make errors when attempting to verify clause satisfaction or when determining the implications of specific variable assignments, particularly misinterpreting the disjunctive nature of clauses or conflating the semantic meaning of positive and negative literals. This direct approach appears to lead to a higher error rate during the reasoning process, as the models struggle to maintain consistency across the complex network of logical constraints represented in the `DIMACS` format.

Conversely, vanilla models like `DeepSeek-V3-0324` and `GPT-4.1`, although performing worse overall, show relatively balanced performance across different formats, indicating lower format sensitivity. These models sometimes employ structured thinking approaches by first translating `DIMACS` inputs into the `Math` format before proceeding with reasoning, or by introducing meaningful symbolic notation during their reasoning process, which appears to enhance reasoning stability. The shorter reasoning chains produced by vanilla models may also contribute to their format robustness, as briefer deductions have fewer opportunities for errors to accumulate. This contrasts with `DeepSeek-R1` and `QwQ-32B`, whose struggles outside the `Math` format seem to stem from relying more on potentially error-prone direct reasoning or trial-and-error within unfamiliar formats, rather than employing systematic format translation or structured analysis.

Achieving AGI likely requires LLMs to reason effectively across diverse formats, enabling the integration of knowledge from different domains and fostering more powerful, generalized reasoning capabilities. `SATQuest` thus serves as a valuable benchmark for assessing LLMs' adaptability and robustness in logical reasoning across various presentation styles.

4 Reinforcement Fine-Tuning

Overview. As demonstrated by the evaluation in Sec. 3, current LLMs exhibit significant limitations in logical reasoning, with notable deficiencies in generalization across *instance scale*, *problem type*, and *question format*. This section explores avenues for enhancing LLM logical reasoning capabilities through Reinforcement Fine-Tuning (RFT), directly utilizing reward signals from the `SATQuest` verifier. Our investigation particularly focuses on two aspects: First, we assess whether `SATQuest`-driven RFT can stimulate LLMs to construct longer reasoning chains, thereby fostering deeper logical deduction. Second, we delve into the relationship between RFT and the generalization deficiencies identified in Sec. 3, aiming to elucidate the specific effects and potential bottlenecks of RFT in improving cross-task and cross-format generalization.

Setup. We select the `Qwen2.5-7B-Instruct` model as the baseline, utilizing the Group Relative Policy Optimization (GRPO) algorithm [Shao and et al., 2024] during fine-tuning. This baseline model is nearly a blank slate, a small vanilla model that only shows marginal performance on `SATSP`-`Math`. The training leverages the 🤗 `sdpkjc/SATQuest-RFT-3k` dataset, comprising $3,000$ CNF instance pairs with $n \in [3, 8]$ variables and clause-to-variable ratios ranging from 2.1 to 4.0. We train

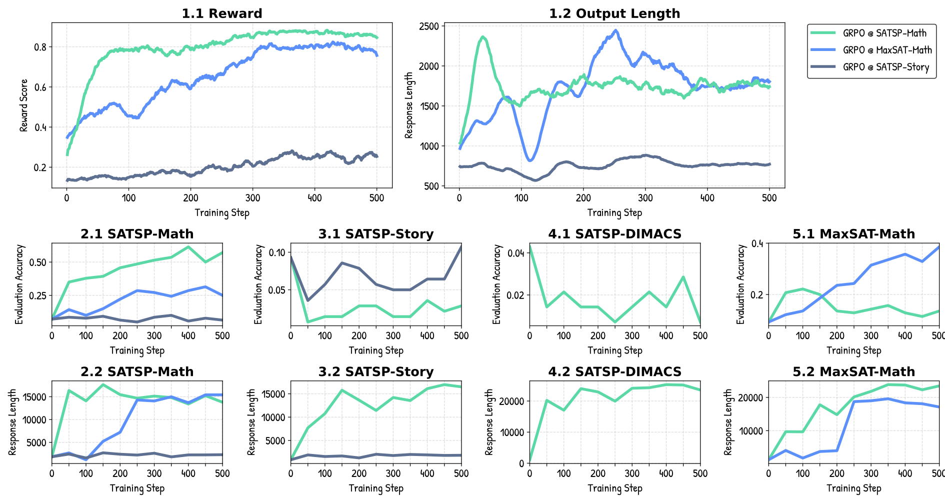

Figure 5: GRPO fine-tuning using SATQuest rewards. Training dynamics (Top: reward, response length) and evaluation performance (Bottom: accuracy, response length on target and generalization tasks) vs. training steps for models fine-tuned on SATSP-Math, MaxSAT-Math, and SATSP-Story.

three distinct models, each focusing on a specific *problem type* and *question format* combination: SATSP-Math, SATSP-Story, MaxSAT-Math.

Our implementation is based on the TRL library [von Werra et al., 2020], adopting the GRPO objective as described in [Shao and et al., 2024]. The prompt template follows the structure proposed in [DeepSeek-AI et al., 2025]. The reward function is designed using the SATQuest verifier, assigning a reward of 1.0 for correct answers and 0.0 for incorrect ones. Additionally, two format correctness rewards, detailed in App. E.1, are incorporated with weights of 0.05 each, complementing the primary reward weight of 1.0. For training we set max_prompt_length to 2048 and $max\_completion\_length$ to 8192, for evaluation we set max_prompt_length + $max\_completion\_length$ to 32768. The training and evaluation parameters are detailed in App. E.2.

Training Dynamics. Training curves (Fig. 5, top row) reveal that models trained on Math-based tasks (SATSP-Math and MaxSAT-Math) achieve higher rewards and generate longer responses than those trained on SATSP-Story. The Math format appears to better facilitate extended reasoning chains that receive positive reinforcement from the verifier. Our SATQuest verifier effectively stimulates extended reasoning development within few training steps, especially with the mathematical format. Response length curves show distinct patterns—rapid initial growth as models learn longer reasoning chains, temporary decline when adapting to format constraints at the training response limit (8192 tokens), followed by stabilization.

Training on SATSP-Story proved less effective, largely due to the baseline model's weak narrative reasoning abilities. While *Logic-RL* [Xie et al., 2025] has successfully stimulated narrative reasoning, our tasks involve substantially higher complexity and scale. Convergence efficiency correlates with task complexity and initial model capabilities, though these factors require further investigation to fully separate.

Generalization Across *Instance Scale and Difficulty*. We observe positive generalization concerning problem complexity. Models fine-tuned on SATSP-Math and MaxSAT-Math demonstrated improved accuracy when evaluated on the corresponding tasks within the evaluation set (Fig. 5, subplots 2.1, 5.1). Crucially, these evaluation instances involved larger scales ($n > 8$) than those used during training ($n \in [3, 8]$), indicating that the learned reasoning skills generalize to more complex instances within the same problem and format.

Generalization Across *Problem Types*. Our results reveal an interesting asymmetry in cross-problem generalization. Fine-tuning on the more complex MaxSAT-Math task led to performance improvements not only on MaxSAT-Math itself but also conferred benefits to the simpler SATSP-Math task. However, the model trained solely on SATSP-Math did not show a corresponding improvement on MaxSAT-Math (compare improvements patterns in Fig. 5, bottom row). This suggests that the reasoning capabilities required for MaxSAT may encompass those needed for SATSP. Strategically, this implies that training on more complex and diverse logical problems could be more effective for fostering robust reasoning skills that generalize to simpler, related problems.

Generalization Across *Question Formats*. Cross-format generalization remains notably difficult. The model fine-tuned on SATSP-Math shows minimal improvement when evaluated on other formats such as SATSP-Story (Fig. 5, subplot 3.1) and even on the structurally similar SATSP-DIMACS task (subplot 4.1). This suggests that reasoning capabilities acquired in the Math format do not readily transfer to logically equivalent tasks presented in alternative formats, whether narrative or machine-readable. Further analysis of failure cases reveals that, after SATSP-Math fine-tuning, the model tends to generate verbose but flawed reasoning when confronted with DIMACS inputs—mirroring the issues described in Sec. 3 and App. C. The model appears to have overfitted to a specific Math-style reasoning pattern, at the expense of its initial structured thinking ability, and fails to effectively translate DIMACS representations into a suitable reasoning form. These findings indicate that small-scale RFT may not be sufficient to overcome format generalization barriers, and that the performance discrepancies across formats observed in Sec. 3 may partially stem from the limitations of format-specific fine-tuning itself.

5 Related Work

Researchers have developed numerous benchmarks to evaluate LLMs' capabilities. Popular evaluations like [Rein et al., 2024, Hendrycks et al., 2021a,b, Srivastava et al., 2023] comprehensively assess LLMs but suffer from data leakage and lack of continuity, often being solved by advanced models within 18 months of introduction. While newer benchmarks [Suzgun et al., 2023, Kazemi et al., 2025, Team et al., 2025, Gema et al., 2025, Wang et al., 2024, Glazer et al., 2024] offer improvements, core issues remain. White et al. [2025] introduced dynamic question banks and automatic scoring but still relies on manual difficulty annotation without multi-dimensional analysis controls. These evaluations primarily assess capabilities rather than providing insights into internal mechanisms. Some studies explore more nuanced approaches: Lin et al. [2025], Xie et al. [2024] use formalized templates with controllable difficulty dimensions for finer-grained analysis. Xie et al. [2025] conducts RL training on *K&K* to investigate how RL enhances reasoning capabilities. He et al. [2024] provides evaluation through multilingual coverage and multi-turn design. Huang et al. [2025], Yu et al. [2025] test mathematical reasoning robustness through minimal perturbations. Research by Hazra et al. [2025] investigates LLM reasoning capabilities through 3-SAT phase transitions. Our work, SATQuest, uses randomly generated CNF instances to prevent data leakage and ensure continuity. We provide five interrelated SAT-based *problem types* and four *question formats* with different information densities. These three orthogonal dimensions—*instance scale*, *problem type*, and *question format*—enable flexible experimental control for future LLM reasoning research.

6 Summary and Limitations

We introduced SATQuest, a novel verifier designed to systematically generate diverse logical reasoning problems from CNF instances, structured along the dimensions of instance scale, problem type, and question format. This tool was employed for both the extensive evaluation of a range of LLMs and for their reinforcement fine-tuning using verifiable rewards derived from PySAT. Our findings highlight significant limitations in current LLMs' logical reasoning, particularly in generalizing beyond familiar mathematical formats and in tackling more complex problem types like MCS and MUS. The key takeaway is that while reinforcement fine-tuning can improve performance on targeted tasks and even generalize to more complex instances of the same type, robust cross-format adaptation remains a substantial hurdle for LLMs. SATQuest proves its value by enabling nuanced, multi-dimensional analysis of LLM reasoning capabilities and facilitating targeted enhancements through its automated problem generation and objective answer verification. Our work is limited by the scale of the reinforcement fine-tuning experiments, which were primarily conducted on a 7B parameter model, and the inherent challenges in achieving broad generalization across all problem types and formats with current RFT techniques. Despite these limitations, SATQuest offers a valuable and systematic framework, serving as a strong starting point for future research aimed at rigorously evaluating and advancing the logical reasoning capabilities of LLMs.

References

OpenAI. OpenAI o3-mini System Card. https://cdn.openai.com/o3-mini-system-card-feb10.pdf, January 2025. Accessed: 2025-04-22.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jiawei Wang, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Shengfeng Ye, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wanjia Zhao, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanhong Xu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025. URL https://arxiv.org/abs/2501.12948.

Qwen Team. Qwq: Reflect deeply on the boundaries of the unknown, November 2024. URL https://qwenlm.github.io/blog/qwq-32b-preview/.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Christopher Berner, Lenny Bogdonoff, Oleg Boiko, Madelaine Boyd, Anna-Luisa Brakman, Greg Brockman, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann, Brittany Carey, Chelsea Carlson, Rory Carmichael, Brooke Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully Chen, Ruby Chen, Jason Chen, Mark Chen, Ben Chess, Chester Cho, Casey Chu, Hyung Won Chung, Dave Cummings, Jeremiah Currier, Yunxing Dai, Cory Decareaux, Thomas Degry, Noah Deutsch, Damien Deville, Arka Dhar, David Dohan, Steve Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti, Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix, Simón Posada Fishman, Juston Forte, Isabella Fulford, Leo Gao, Elie Georges, Christian Gibson, Vik Goel, Tarun Gogineni, Gabriel

Goh, Rapha Gontijo-Lopes, Jonathan Gordon, Morgan Grafstein, Scott Gray, Ryan Greene, Joshua Gross, Shixiang Shane Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff Harris, Yuchen He, Mike Heaton, Johannes Heidecke, Chris Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele, Brandon Houghton, Kenny Hsu, Shengli Hu, Xin Hu, Joost Huizinga, Shantanu Jain, Shawn Jain, Joanne Jang, Angela Jiang, Roger Jiang, Haozhun Jin, Denny Jin, Shino Jomoto, Billie Jonn, Heewoo Jun, Tomer Kaftan, Łukasz Kaiser, Ali Kamali, Ingmar Kanitscheider, Nitish Shirish Keskar, Tabarak Khan, Logan Kilpatrick, Jong Wook Kim, Christina Kim, Yongjik Kim, Jan Hendrik Kirchner, Jamie Kiros, Matt Knight, Daniel Kokotajlo, Łukasz Kondraciuk, Andrew Kondrich, Aris Konstantinidis, Kyle Kosic, Gretchen Krueger, Vishal Kuo, Michael Lampe, Ikai Lan, Teddy Lee, Jan Leike, Jade Leung, Daniel Levy, Chak Ming Li, Rachel Lim, Molly Lin, Stephanie Lin, Mateusz Litwin, Theresa Lopez, Ryan Lowe, Patricia Lue, Anna Makanju, Kim Malfacini, Sam Manning, Todor Markov, Yaniv Markovski, Bianca Martin, Katie Mayer, Andrew Mayne, Bob McGrew, Scott Mayer McKinney, Christine McLeavey, Paul McMillan, Jake McNeil, David Medina, Aalok Mehta, Jacob Menick, Luke Metz, Andrey Mishchenko, Pamela Mishkin, Vinnie Monaco, Evan Morikawa, Daniel Mossing, Tong Mu, Mira Murati, Oleg Murk, David Mély, Ashvin Nair, Reiichiro Nakano, Rajeev Nayak, Arvind Neelakantan, Richard Ngo, Hyeonwoo Noh, Long Ouyang, Cullen O'Keefe, Jakub Pachocki, Alex Paino, Joe Palermo, Ashley Pantuliano, Giambattista Parascandolo, Joel Parish, Emy Parparita, Alex Passos, Mikhail Pavlov, Andrew Peng, Adam Perelman, Filipe de Avila Belbute Peres, Michael Petrov, Henrique Ponde de Oliveira Pinto, Michael, Pokorny, Michelle Pokrass, Vitchyr H. Pong, Tolly Powell, Alethea Power, Boris Power, Elizabeth Proehl, Raul Puri, Alec Radford, Jack Rae, Aditya Ramesh, Cameron Raymond, Francis Real, Kendra Rimbach, Carl Ross, Bob Rotsted, Henri Roussez, Nick Ryder, Mario Saltarelli, Ted Sanders, Shibani Santurkar, Girish Sastry, Heather Schmidt, David Schnurr, John Schulman, Daniel Selsam, Kyla Sheppard, Toki Sherbakov, Jessica Shieh, Sarah Shoker, Pranav Shyam, Szymon Sidor, Eric Sigler, Maddie Simens, Jordan Sitkin, Katarina Slama, Ian Sohl, Benjamin Sokolowsky, Yang Song, Natalie Staudacher, Felipe Petroski Such, Natalie Summers, Ilya Sutskever, Jie Tang, Nikolas Tezak, Madeleine B. Thompson, Phil Tillet, Amin Tootoonchian, Elizabeth Tseng, Preston Tuggle, Nick Turley, Jerry Tworek, Juan Felipe Cerón Uribe, Andrea Vallone, Arun Vijayvergiya, Chelsea Voss, Carroll Wainwright, Justin Jay Wang, Alvin Wang, Ben Wang, Jonathan Ward, Jason Wei, CJ Weinmann, Akila Welihinda, Peter Welinder, Jiayi Weng, Lilian Weng, Matt Wiethoff, Dave Willner, Clemens Winter, Samuel Wolrich, Hannah Wong, Lauren Workman, Sherwin Wu, Jeff Wu, Michael Wu, Kai Xiao, Tao Xu, Sarah Yoo, Kevin Yu, Qiming Yuan, Wojciech Zaremba, Rowan Zellers, Chong Zhang, Marvin Zhang, Shengjia Zhao, Tianhao Zheng, Juntang Zhuang, William Zhuk, and Barret Zoph. Gpt-4 technical report, 2024. URL https://arxiv.org/abs/2303.08774.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed H. Chi, Quoc V Le, and Denny Zhou. Chain of thought prompting elicits reasoning in large language models. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=_VjQlMeSB_J.

Bill Yuchen Lin, Ronan Le Bras, Kyle Richardson, Ashish Sabharwal, Radha Poovendran, Peter Clark, and Yejin Choi. Zebralogic: On the scaling limits of llms for logical reasoning, 2025. URL https://arxiv.org/abs/2502.01100.

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. In *The 4th Workshop on Mathematical Reasoning and AI at NeurIPS'24*, 2024. URL https://openreview.net/forum?id=mxX8WdPCx9.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the MATH dataset. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021a. URL https://openreview.net/forum?id=7Bywt2mQsCe.

Colin White, Samuel Dooley, Manley Roberts, Arka Pal, Benjamin Feuer, Siddhartha Jain, Ravid Shwartz-Ziv, Neel Jain, Khalid Saifullah, Sreemanti Dey, Shubh-Agrawal, Sandeep Singh Sandha, Siddartha Venkat Naidu, Chinmay Hegde, Yann LeCun, Tom Goldstein, Willie Neiswanger, and Micah Goldblum. Livebench: A challenging, contamination-limited LLM benchmark. In

The Thirteenth International Conference on Learning Representations, 2025. URL https://openreview.net/forum?id=sKYHBTAxVa.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. GPQA: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=Ti67584b98.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations*, 2021b. URL https://openreview.net/forum?id=d7KBjmI3GmQ.

Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R. Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, Agnieszka Kluska, Aitor Lewkowycz, Akshat Agarwal, Alethea Power, Alex Ray, Alex Warstadt, Alexander W. Kocurek, Ali Safaya, Ali Tazarv, Alice Xiang, Alicia Parrish, Allen Nie, Aman Hussain, Amanda Askell, Amanda Dsouza, Ambrose Slone, Ameet Rahane, Anantharaman S. Iyer, Anders Johan Andreassen, Andrea Madotto, Andrea Santilli, Andreas Stuhlmüller, Andrew M. Dai, Andrew La, Andrew Kyle Lampinen, Andy Zou, Angela Jiang, Angelica Chen, Anh Vuong, Animesh Gupta, Anna Gottardi, Antonio Norelli, Anu Venkatesh, Arash Gholamidavoodi, Arfa Tabassum, Arul Menezes, Arun Kirubarajan, Asher Mullokandov, Austin Herrick, Avia Efrat, Aykut Erdem, Ayla Karakaş, B. Ryan Roberts, Bao Sheng Loe, Barret Zoph, Bartłomiej Bojanowski, Batuhan Özyurt, Behnam Hedayatnia, Behnam Neyshabur, Benjamin Inden, Benno Stein, Berk Ekmekci, Bill Yuchen Lin, Blake Howald, Bryan Orinion, Cameron Diao, Cameron Dour, Catherine Stinson, Cedrick Argueta, Cesar Ferri, Chandan Singh, Charles Rathkopf, Chenlin Meng, Chitta Baral, Chiyu Wu, Chris Callison-Burch, Christopher Waites, Christian Voigt, Christopher D Manning, Christopher Potts, Cindy Ramirez, Clara E. Rivera, Clemencia Siro, Colin Raffel, Courtney Ashcraft, Cristina Garbacea, Damien Sileo, Dan Garrette, Dan Hendrycks, Dan Kilman, Dan Roth, C. Daniel Freeman, Daniel Khashabi, Daniel Levy, Daniel Moseguí González, Danielle Perszyk, Danny Hernandez, Danqi Chen, Daphne Ippolito, Dar Gilboa, David Dohan, David Drakard, David Jurgens, Debajyoti Datta, Deep Ganguli, Denis Emelin, Denis Kleyko, Deniz Yuret, Derek Chen, Derek Tam, Dieuwke Hupkes, Diganta Misra, Dilyar Buzan, Dimitri Coelho Mollo, Diyi Yang, Dong-Ho Lee, Dylan Schrader, Ekaterina Shutova, Ekin Dogus Cubuk, Elad Segal, Eleanor Hagerman, Elizabeth Barnes, Elizabeth Donoway, Ellie Pavlick, Emanuele Rodolà, Emma Lam, Eric Chu, Eric Tang, Erkut Erdem, Ernie Chang, Ethan A Chi, Ethan Dyer, Ethan Jerzak, Ethan Kim, Eunice Engefu Manyasi, Evgenii Zheltonozhskii, Fanyue Xia, Fatemeh Siar, Fernando Martínez-Plumed, Francesca Happé, Francois Chollet, Frieda Rong, Gaurav Mishra, Genta Indra Winata, Gerard de Melo, Germàn Kruszewski, Giambattista Parascandolo, Giorgio Mariani, Gloria Xinyue Wang, Gonzalo Jaimovitch-Lopez, Gregor Betz, Guy Gur-Ari, Hana Galijasevic, Hannah Kim, Hannah Rashkin, Hannaneh Hajishirzi, Harsh Mehta, Hayden Bogar, Henry Francis Anthony Shevlin, Hinrich Schuetze, Hiromu Yakura, Hongming Zhang, Hugh Mee Wong, Ian Ng, Isaac Noble, Jaap Jumelet, Jack Geissinger, Jackson Kernion, Jacob Hilton, Jaehoon Lee, Jaime Fernández Fisac, James B Simon, James Koppel, James Zheng, James Zou, Jan Kocon, Jana Thompson, Janelle Wingfield, Jared Kaplan, Jarema Radom, Jascha Sohl-Dickstein, Jason Phang, Jason Wei, Jason Yosinski, Jekaterina Novikova, Jelle Bosscher, Jennifer Marsh, Jeremy Kim, Jeroen Taal, Jesse Engel, Jesujoba Alabi, Jiacheng Xu, Jiaming Song, Jillian Tang, Joan Waweru, John Burden, John Miller, John U. Balis, Jonathan Batchelder, Jonathan Berant, Jörg Frohberg, Jos Rozen, Jose Hernandez-Orallo, Joseph Boudeman, Joseph Guerr, Joseph Jones, Joshua B. Tenenbaum, Joshua S. Rule, Joyce Chua, Kamil Kanclerz, Karen Livescu, Karl Krauth, Karthik Gopalakrishnan, Katerina Ignatyeva, Katja Markert, Kaustubh Dhole, Kevin Gimpel, Kevin Omondi, Kory Wallace Mathewson, Kristen Chiafullo, Ksenia Shkaruta, Kumar Shridhar, Kyle McDonell, Kyle Richardson, Laria Reynolds, Leo Gao, Li Zhang, Liam Dugan, Lianhui Qin, Lidia Contreras-Ochando, Louis-Philippe Morency, Luca Moschella, Lucas Lam, Lucy Noble, Ludwig Schmidt, Luheng He, Luis Oliveros-Colón, Luke Metz, Lütfi Kerem Senel, Maarten Bosma, Maarten Sap, Maartje Ter Hoeve, Maheen Farooqi, Manaal Faruqui, Mantas Mazeika, Marco Baturan, Marco Marelli, Marco Maru, Maria Jose Ramirez-Quintana, Marie Tolkiehn, Mario Giulianelli, Martha Lewis, Martin Potthast, Matthew L Leavitt, Matthias Hagen, Mátyás Schubert, Medina Orduna Baitemirova, Melody Arnaud, Melvin McElrath, Michael Andrew Yee, Michael Cohen, Michael Gu, Michael Ivanitskiy, Michael Starritt, Michael Strube, Michał Swędrowski, Michele Bevilacqua, Michihiro Yasunaga, Mihir Kale, Mike Cain, Mimee Xu, Mirac

Suzgun, Mitch Walker, Mo Tiwari, Mohit Bansal, Moin Aminnaseri, Mor Geva, Mozhdeh Gheini, Mukund Varma T, Nanyun Peng, Nathan Andrew Chi, Nayeon Lee, Neta Gur-Ari Krakover, Nicholas Cameron, Nicholas Roberts, Nick Doiron, Nicole Martinez, Nikita Nangia, Niklas Deckers, Niklas Muennighoff, Nitish Shirish Keskar, Niveditha S. Iyer, Noah Constant, Noah Fiedel, Nuan Wen, Oliver Zhang, Omar Agha, Omar Elbaghdadi, Omer Levy, Owain Evans, Pablo Antonio Moreno Casares, Parth Doshi, Pascale Fung, Paul Pu Liang, Paul Vicol, Pegah Alipoormolabashi, Peiyuan Liao, Percy Liang, Peter W Chang, Peter Eckersley, Phu Mon Htut, Pinyu Hwang, Piotr Miłkowski, Piyush Patil, Pouya Pezeshkpour, Priti Oli, Qiaozhu Mei, Qing Lyu, Qinlang Chen, Rabin Banjade, Rachel Etta Rudolph, Raefer Gabriel, Rahel Habacker, Ramon Risco, Raphaël Millière, Rhythm Garg, Richard Barnes, Rif A. Saurous, Riku Arakawa, Robbe Raymaekers, Robert Frank, Rohan Sikand, Roman Novak, Roman Sitelew, Ronan Le Bras, Rosanne Liu, Rowan Jacobs, Rui Zhang, Russ Salakhutdinov, Ryan Andrew Chi, Seungjae Ryan Lee, Ryan Stovall, Ryan Teehan, Rylan Yang, Sahib Singh, Saif M. Mohammad, Sajant Anand, Sam Dillavou, Sam Shleifer, Sam Wiseman, Samuel Gruetter, Samuel R. Bowman, Samuel Stern Schoenholz, Sanghyun Han, Sanjeev Kwatra, Sarah A. Rous, Sarik Ghazarian, Sayan Ghosh, Sean Casey, Sebastian Bischoff, Sebastian Gehrmann, Sebastian Schuster, Sepideh Sadeghi, Shadi Hamdan, Sharon Zhou, Shashank Srivastava, Sherry Shi, Shikhar Singh, Shima Asaadi, Shixiang Shane Gu, Shubh Pachchigar, Shubham Toshniwal, Shyam Upadhyay, Shyamolima Shammie Debnath, Siamak Shakeri, Simon Thormeyer, Simone Melzi, Siva Reddy, Sneha Priscilla Makini, Soo-Hwan Lee, Spencer Torene, Sriharsha Hatwar, Stanislas Dehaene, Stefan Divic, Stefano Ermon, Stella Biderman, Stephanie Lin, Stephen Prasad, Steven Piantadosi, Stuart Shieber, Summer Misherghi, Svetlana Kiritchenko, Swaroop Mishra, Tal Linzen, Tal Schuster, Tao Li, Tao Yu, Tariq Ali, Tatsunori Hashimoto, Te-Lin Wu, Théo Desbordes, Theodore Rothschild, Thomas Phan, Tianle Wang, Tiberius Nkinyili, Timo Schick, Timofei Kornev, Titus Tunduny, Tobias Gerstenberg, Trenton Chang, Trishala Neeraj, Tushar Khot, Tyler Shultz, Uri Shaham, Vedant Misra, Vera Demberg, Victoria Nyamai, Vikas Raunak, Vinay Venkatesh Ramasesh, vinay uday prabhu, Vishakh Padmakumar, Vivek Srikumar, William Fedus, William Saunders, William Zhang, Wout Vossen, Xiang Ren, Xiaoyu Tong, Xinran Zhao, Xinyi Wu, Xudong Shen, Yadollah Yaghoobzadeh, Yair Lakretz, Yangqiu Song, Yasaman Bahri, Yejin Choi, Yichi Yang, Sophie Hao, Yifu Chen, Yonatan Belinkov, Yu Hou, Yufang Hou, Yuntao Bai, Zachary Seid, Zhuoye Zhao, Zijian Wang, Zijie J. Wang, Zirui Wang, and Ziyi Wu. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856. URL https://openreview.net/forum?id=uyTL5Bvosj. Featured Certification.

Alexey Ignatiev, Antonio Morgado, and Joao Marques-Silva. PySAT: A Python toolkit for prototyping with SAT oracles. In *SAT*, pages 428–437, 2018. doi: 10.1007/978-3-319-94144-8_26. URL https://doi.org/10.1007/978-3-319-94144-8_26.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V. Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, Yuling Gu, Saumya Malik, Victoria Graf, Jena D. Hwang, Jiangjiang Yang, Ronan Le Bras, Oyvind Tafjord, Chris Wilhelm, Luca Soldaini, Noah A. Smith, Yizhong Wang, Pradeep Dasigi, and Hannaneh Hajishirzi. Tulu 3: Pushing frontiers in open language model post-training, 2025. URL https://arxiv.org/abs/2411.15124.

Philipp Mondorf and Barbara Plank. Liar, liar, logical mire: A benchmark for suppositional reasoning in large language models. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 7114–7137, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.404. URL https://aclanthology.org/2024.emnlp-main.404/.

Gilles Audemard and Laurent Simon. On the glucose sat solver. *International Journal on Artificial Intelligence Tools*, 27(01):1840001, 2018. doi: 10.1142/S0218213018400018. URL https://doi.org/10.1142/S0218213018400018.

Alexey Ignatiev, Antonio Morgado, and Joao Marques-Silva. Rc2: an efficient maxsat solver. *Journal on Satisfiability, Boolean Modelling and Computation*, 11(1):53–64, 2019. doi: 10.3233/SAT190116. URL https://journals.sagepub.com/doi/abs/10.3233/SAT190116.

Carlos Mencía, Alessandro Previti, and Joao Marques-Silva. Literal-based mcs extraction. In *Proceedings of the 24th International Conference on Artificial Intelligence*, IJCAI'15, page 1973–1979. AAAI Press, 2015. ISBN 9781577357384.

Joao Marques-Silva. Minimal unsatisfiability: Models, algorithms and applications (invited paper). In *Proceedings of the 2010 40th IEEE International Symposium on Multiple-Valued Logic*, ISMVL '10, page 9–14, USA, 2010. IEEE Computer Society. ISBN 9780769540245. doi: 10.1109/ISMVL.2010.11. URL https://doi.org/10.1109/ISMVL.2010.11.

Zhihong Shao and et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Leandro von Werra, Younes Belkada, Lewis Tunstall, Edward Beeching, Tristan Thrush, Nathan Lambert, Shengyi Huang, Kashif Rasul, and Quentin Gallouédec. Trl: Transformer reinforcement learning. https://github.com/huggingface/trl, 2020.

Tian Xie, Zitian Gao, Qingnan Ren, Haoming Luo, Yuqian Hong, Bryan Dai, Joey Zhou, Kai Qiu, Zhirong Wu, and Chong Luo. Logic-rl: Unleashing llm reasoning with rule-based reinforcement learning, 2025. URL https://arxiv.org/abs/2502.14768.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc Le, Ed Chi, Denny Zhou, and Jason Wei. Challenging BIG-bench tasks and whether chain-of-thought can solve them. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, *Findings of the Association for Computational Linguistics: ACL 2023*, pages 13003–13051, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-acl.824. URL https://aclanthology.org/2023.findings-acl.824/.

Mehran Kazemi, Bahare Fatemi, Hritik Bansal, John Palowitch, Chrysovalantis Anastasiou, Sanket Vaibhav Mehta, Lalit K. Jain, Virginia Aglietti, Disha Jindal, Peter Chen, Nishanth Dikkala, Gladys Tyen, Xin Liu, Uri Shalit, Silvia Chiappa, Kate Olszewska, Yi Tay, Vinh Q. Tran, Quoc V. Le, and Orhan Firat. Big-bench extra hard, 2025. URL https://arxiv.org/abs/2502.19187.

P Team, Xinrun Du, Yifan Yao, Kaijing Ma, Bingli Wang, Tianyu Zheng, King Zhu, Minghao Liu, Yiming Liang, Xiaolong Jin, Zhenlin Wei, Chujie Zheng, Kaixin Deng, Shawn Gavin, Shian Jia, Sichao Jiang, Yiyan Liao, Rui Li, Qinrui Li, Sirun Li, Yizhi Li, Yunwen Li, David Ma, Yuansheng Ni, Haoran Que, Qiyao Wang, Zhoufutu Wen, Siwei Wu, Tyshawn Hsing, Ming Xu, Zhenzhu Yang, Zekun Moore Wang, Junting Zhou, Yuelin Bai, Xingyuan Bu, Chenglin Cai, Liang Chen, Yifan Chen, Chengtuo Cheng, Tianhao Cheng, Keyi Ding, Siming Huang, Yun Huang, Yaoru Li, Yizhe Li, Zhaoqun Li, Tianhao Liang, Chengdong Lin, Hongquan Lin, Yinghao Ma, Tianyang Pang, Zhongyuan Peng, Zifan Peng, Qige Qi, Shi Qiu, Xingwei Qu, Shanghaoran Quan, Yizhou Tan, Zili Wang, Chenqing Wang, Hao Wang, Yiya Wang, Yubo Wang, Jiajun Xu, Kexin Yang, Ruibin Yuan, Yuanhao Yue, Tianyang Zhan, Chun Zhang, Jinyang Zhang, Xiyue Zhang, Xingjian Zhang, Yue Zhang, Yongchi Zhao, Xiangyu Zheng, Chenghua Zhong, Yang Gao, Zhoujun Li, Dayiheng Liu, Qian Liu, Tianyu Liu, Shiwen Ni, Junran Peng, Yujia Qin, Wenbo Su, Guoyin Wang, Shi Wang, Jian Yang, Min Yang, Meng Cao, Xiang Yue, Zhaoxiang Zhang, Wangchunshu Zhou, Jiaheng Liu, Qunshu Lin, Wenhao Huang, and Ge Zhang. Supergpqa: Scaling llm evaluation across 285 graduate disciplines, 2025. URL https://arxiv.org/abs/2502.14739.

Aryo Pradipta Gema, Joshua Ong Jun Leang, Giwon Hong, Alessio Devoto, Alberto Carlo Maria Mancino, Rohit Saxena, Xuanli He, Yu Zhao, Xiaotang Du, Mohammad Reza Ghasemi Madani, Claire Barale, Robert McHardy, Joshua Harris, Jean Kaddour, Emile Van Krieken, and Pasquale Minervini. Are we done with MMLU? In Luis Chiruzzo, Alan Ritter, and Lu Wang, editors, *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 5069–5096, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. ISBN 979-8-89176-189-6. URL https://aclanthology.org/2025.naacl-long.262/.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, Tianle Li, Max Ku, Kai Wang, Alex Zhuang, Rongqi Fan, Xiang Yue, and Wenhu Chen. MMLU-pro: A more robust and challenging multi-task language understanding benchmark. In *The Thirty-eight Conference on Neural Information Processing*

Systems Datasets and Benchmarks Track, 2024. URL https://openreview.net/forum?id=y10DM6R2r3.

Elliot Glazer, Ege Erdil, Tamay Besiroglu, Diego Chicharro, Evan Chen, Alex Gunning, Caroline Falkman Olsson, Jean-Stanislas Denain, Anson Ho, Emily de Oliveira Santos, Olli Järviniemi, Matthew Barnett, Robert Sandler, Matej Vrzala, Jaime Sevilla, Qiuyu Ren, Elizabeth Pratt, Lionel Levine, Grant Barkley, Natalie Stewart, Bogdan Grechuk, Tetiana Grechuk, Shreepranav Varma Enugandla, and Mark Wildon. Frontiermath: A benchmark for evaluating advanced mathematical reasoning in ai, 2024. URL https://arxiv.org/abs/2411.04872.

Yun He, Di Jin, Chaoqi Wang, Chloe Bi, Karishma Mandyam, Hejia Zhang, Chen Zhu, Ning Li, Tengyu Xu, Hongjiang Lv, Shruti Bhosale, Chenguang Zhu, Karthik Abinav Sankararaman, Eryk Helenowski, Melanie Kambadur, Aditya Tayade, Hao Ma, Han Fang, and Sinong Wang. Multi-if: Benchmarking llms on multi-turn and multilingual instructions following, 2024. URL https://arxiv.org/abs/2410.15553.

Shulin Huang, Linyi Yang, Yan Song, Shuang Chen, Leyang Cui, Ziyu Wan, Qingcheng Zeng, Ying Wen, Kun Shao, Weinan Zhang, Jun Wang, and Yue Zhang. Thinkbench: Dynamic out-of-distribution evaluation for robust llm reasoning, 2025. URL https://arxiv.org/abs/2502.16268.

Tong Yu, Yongcheng Jing, Xikun Zhang, Wentao Jiang, Wenjie Wu, Yingjie Wang, Wenbin Hu, Bo Du, and Dacheng Tao. Benchmarking reasoning robustness in large language models, 2025. URL https://arxiv.org/abs/2503.04550.

Rishi Hazra, Gabriele Venturato, Pedro Zuidberg Dos Martires, and Luc De Raedt. Have large language models learned to reason? a characterization via 3-sat phase transition, 2025. URL https://arxiv.org/abs/2504.03930.

Daniel Selsam, Matthew Lamm, Benedikt Bünz, Percy Liang, Leonardo de Moura, and David L. Dill. Learning a SAT solver from single-bit supervision. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=HJMC_iA5tm.

A Generate CNF Pair

The CNF generation algorithm is adapted from [Selsam et al., 2019][3].

Algorithm 1 Generate CNF Pair (Satisfiable and Unsatisfiable)

procedure GenCNFPair(n, m, p_{k_2}, p_{geo})
  $unsat\_clauses \leftarrow \emptyset$
  while $solve(unsat\_clauses)$ **do** ▷ Generate unsatisfiable CNF
    $unsat\_clauses \leftarrow \emptyset$
    while $|unsat| < m$ **do**
      $k \leftarrow 1$ if $rand() < p_{k_2}$ else $2 + Geometric(p_{geo})$
      $k \leftarrow \min(k, n)$
      $clause \leftarrow RandomClause(n, k)$ ▷ Random literals with polarity
      $unsat\_clauses \leftarrow unsat\_clauses \cup \{clause\}$
    end while
  end while
  $sat\_clauses \leftarrow unsat\_clauses$
  while $\neg solve(sat\_clauses)$ **do** ▷ Convert to satisfiable CNF
    $sat\_clauses \leftarrow RandomFlipClause(sat\_clauses)$ ▷ Flip literal polarities
  end while
  return $unsat\_clauses, sat\_clauses$
end procedure

[3] https://github.com/dselsam/neurosat/blob/master/python/gen_sr_dimacs.py

B Prompt Details

In this appendix, we present the complete prompt templates and representative example outputs used to instruct LLMs to produce binary string responses for SAT-based reasoning tasks. These templates specify the precise wording, formatting requirements, and task descriptions used during both evaluation and reinforcement fine-tuning phases. Our prompt construction strategy adapts and extends established templates from OpenAI's `simple-eval`[4] and `DeepSeek-R1` [DeepSeek-AI et al., 2025]. The following subsections detail how we systematically assemble task prompts and incorporate system-level configurations for both evaluation (Sec. 3) and fine-tuning (Sec. 4) experiments. Figure 6 illustrates the general structure of the task prompt used throughout our experiments.

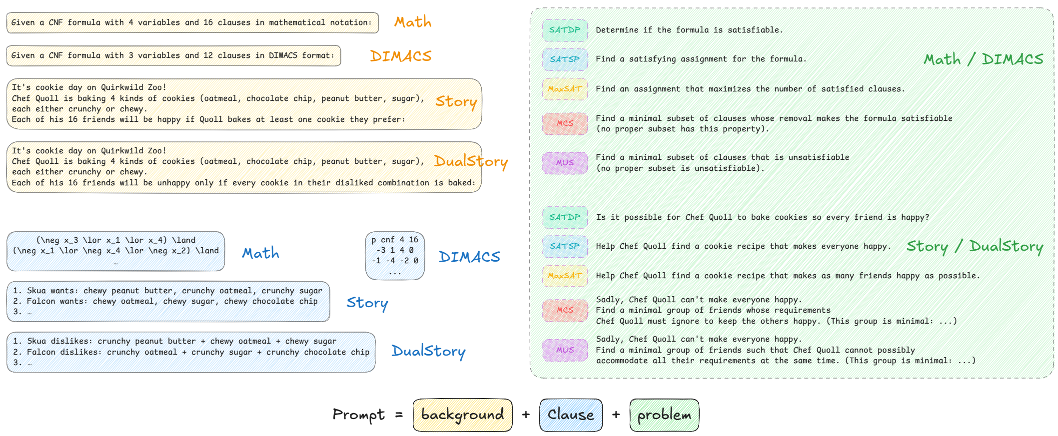

Figure 6: Task Prompt Construction in `SATQuest`

B.1 Evaluation Prompt Template

In Sec. 3, we employ the following prompt template that includes Chain-of-Thought (CoT) guidance and specific output format instructions to ensure consistent responses across all models.

For vanilla model, add system prompt: `"You are a helpful assistant."`.

```
Solve the following problem step by step. The last line of your response should be of the form Answer:
    ↪ $ANSWER (without quotes) where $ANSWER is the answer to the problem.

Given a CNF formula with 4 variables and 16 clauses in DIMACS format:

p cnf 4 16
-3 1 4 0
-1 -4 -2 0
-1 4 2 0
2 -1 -4 0
2 -3 4 0
-3 -4 1 0
-4 1 -3 0
1 2 -4 0
-3 -2 1 0
4 -3 1 0
-1 -3 2 0
2 -3 4 0
-1 -2 3 0
2 3 4 0
2 3 1 0
1 3 -4 0

Find a satisfying assignment for the formula.
Output a binary string of length 4 ('1' for true, '0' for false).

Remember to put your answer on its own line after "Answer:", and you do not need to use a \boxed command.
```

[4] https://github.com/openai/simple-evals/blob/0f2cf3/math_eval.py

B.2 Reinforcement Fine-Tuning Prompt Template

For Sec. 4, we adapt the prompt to encourage explicit CoT reasoning by combining system instructions with a structured user message format.

```
<|im_start|>system
You are a helpful AI Assistant that provides well-reasoned and detailed responses. You first think about
      ↪ the reasoning process as an internal monologue and then provide the user with the answer. Respond
      ↪ in the following format: <think>\n...\n</think>\n<answer>\n...\n</answer>
<|im_end|>
<|im_start|>user
Given a CNF formula with 4 variables and 16 clauses in DIMACS format:

p cnf 4 16
-3 1 4 0
-1 -4 -2 0
-1 4 2 0
2 -1 -4 0
2 -3 4 0
-3 -4 1 0
-4 1 -3 0
1 2 -4 0
-3 -2 1 0
4 -3 1 0
-1 -3 2 0
2 -3 4 0
-1 -2 3 0
2 3 4 0
2 3 1 0
1 3 -4 0

Find a satisfying assignment for the formula.
Output a binary string of length 4 ('1' for true, '0' for false).

Show your work in <think> </think> tags. And return the final answer in <answer> </answer> tags, for
      ↪ example <answer> 0101 </answer>.
<|im_end|>
<|im_start|>assistant
```

C Case Studies

Our analysis reveals that both DeepSeek-R1 and QwQ-32B predominantly reason directly with the DIMACS format rather than translating it into formal mathematical notation. Throughout the four case studies, segments highlighted in green demonstrate instances where the models correctly comprehend the CNF structure, especially regarding the disjunctive (OR) semantics within clauses. Conversely, red-highlighted sections expose critical reasoning flaws: models frequently misinterpret the disjunctive nature of clauses or conflate the semantic meaning of positive and negative literals (representing true versus false values). These observations suggest that direct reasoning within the DIMACS representation, without proper symbolic abstraction, frequently results in semantic drift and format-specific reasoning errors.

Case1: DeepSeek-R1 @ SATSP-DIMACS

```
Okay, so I need to find a satisfying assignment for this CNF formula with 9 variables and 36 clauses.

...

So, for each clause, at least one of the literals must be true in the assignment.

...

For each variable, check if all occurrences are positive, all negative, or mixed.

Let's start with variable 1:

Looking through the clauses where variable 1 appears.

Clause 5: -1 2 -6 5 8. So -1 here.

Clause 6:8 1 9 -6. Positive 1.

...

Clause1: -2 -9 6 -> with 6=0, this becomes -2 AND -9 -> so both must be true. So variables2 and 9 must be 0.

...

All clauses seem to be satisfied. So the assignment is valid.
```

Full prompt and response: https://gist.github.com/sdpkjc/3e3df5e9f268588dab862e795b080258

Case2: DeepSeek-R1 @ SATSP-DIMACS

```
Okay, so I need to find a satisfying assignment for this CNF formula with 10 variables and 40 clauses.
     ↪ Hmm, let's think. The variables are numbered 1 through 10, and each line in the DIMACS format
     ↪ represents a clause.
The clauses are ORs of literals, and the entire formula is the AND of all these clauses.
So the assignment has to make at least one literal in each clause true.

...

Clause 5: -1 -> negative occurrence.

Clause 6: 1 -> positive.

...

Ok, moving on.

Clause28:-5 -1 -> both must be false. So variable5=0 and variable1=0.

Wait, ...

All other clauses seem to have been handled earlier. So this assignment should work. Thus, the binary
     ↪ string is 0001010100.
```

Full prompt and response: https://gist.github.com/sdpkjc/ab0963d08dc65cb9352e4fa489b39040

Case3: QwQ-32B @ SATSP-DIMACS

```
Okay, so I have this CNF formula with 9 variables and 36 clauses, and I need to find a satisfying
    ↪ assignment.

...

Let me start by writing down some of the clauses:

1. 5 3 -4 9 0 -> variables 5,3 are true, 4 is false, 9 is true.
2. -1 -2 7 -6 0 -> variables 1 and 2 are false, 7 and 6 are false?
Wait, no, the clause is satisfied if at least one literal is true.
So for this clause, either -1 (1 is false), -2 (2 false), 7 (true), or -6 (6 false). So ...

...

So this should be a valid assignment.
```

Full prompt and response: https://gist.github.com/sdpkjc/415a9f0d6e9166f9ca6c19f4bbb5be4f

Case4: QwQ-32B @ SATSP-DIMACS

```
Okay, so I need to find a satisfying assignment for this CNF formula with 10 variables and 40 clauses.
    ↪ Hmm, that's a bit of a challenge.
Let me think about how to approach this.
Since it's a CNF formula, each clause has to be satisfied by at least one literal being true.

...

Looking at the clauses:

The first few clauses are:

-9 4 -10 0 -> clause 1

6 -2 -8 -1 0 -> clause 2

....

Clause4: -1 -2 -4 -> since variables1 and2 are 0, this clause requires -4 (so variable4 must be 0).

...

Yes, that works.

Thus, the satisfying assignment is 1000101101.
```

Full prompt and response: https://gist.github.com/sdpkjc/2aef69a7b663be4e0ca5325b7526ab5d

D Evaluation Configs

In all experiments in Sec. 3, we use APIs for evaluation. The specific model APIs and IDs are listed in Table 1, and the parameters for vanilla models are shown in Table 2. For reasoning models, we use their default configurations.

Table 1: Model API / ID of our Evaluated Models.

Model Name	API / ID
o3-mini	Azure API[5]: o3-mini-2025-01-31
GPT-4.1	Azure API: gpt-4.1-2025-04-14
DeepSeek-R1	VolcEngine API[6]: deepseek-r1
DeepSeek-V3-0324	VolcEngine API: deepseek-v3-0324
DeepSeek-R1-Distill-Qwen-7B	VolcEngine API: deepseek-r1-distill-qwen-7b
DeepSeek-R1-Distill-Qwen-32B	VolcEngine API: deepseek-r1-distill-qwen-32b
QwQ-32B	Alibaba Cloud API: qwq-32b-plus[7]
Qwen2.5-7B-Instruct	Alibaba Cloud API: qwen2.5-7b-instruct
Qwen2.5-32B-Instruct	Alibaba Cloud API: qwen2.5-32b-instruct

Table 2: Evaluation Parameters for Vanilla Models

Parameter	Value
temperature	0.6
top_p	1.0
max_tokens	16384

[5] *Azure OpenAI API*: https://azure.microsoft.com/en-us/products/ai-services/openai-service
[6] *ByteDance VolcEngine AI platform API*: https://www.volcengine.com/
[7] *Alibaba Cloud API*: https://www.alibabacloud.com/en

E Reinforcement Fine-Tuning Configs

E.1 Format Reward Functions

The format reward functions are adapted from the `huggingface/Open-R1` library[8].

```
def tag_count_reward(completions, **kwargs) -> list[float]:
    def count_tags(text: str) -> float:
        count = 0.0
        if text.count("<think>") == 1:
            count += 0.25
        if text.count("</think>") == 1:
            count += 0.25
        if text.count("<answer>") == 1:
            count += 0.25
        if text.count("</answer>") == 1:
            count += 0.25
        return count

    contents = [completion[0]["content"] for completion in completions]
    return [count_tags(c) for c in contents]

def format_reward(completions, **kwargs):
    _PATTERN = re.compile(r"<think>.*?</think>\s?<answer>.*?</answer>", flags=re.DOTALL)
    completion_contents = [completion[0]["content"] for completion in completions]
    rewards = []
    for c in completion_contents:
        text = str(c)
        total_len = len(text)
        if total_len == 0:
            rewards.append(0.0)
            continue

        m = _PATTERN.search(text)
        match_len = len(m.group()) if m else 0
        rewards.append(match_len / total_len)

    return rewards
```

E.2 Training Parameters

The training parameters used for GRPO of the `Qwen2.5-7B-Instruct` model are summarized in Table 3.

Table 3: Training Parameters for GRPO

Parameter	**Value**
`learning_rate`	0.000002
`batch_size`	`num_generations` $\times 8 = 128$
`max_grad_norm`	0.3
`num_iterations`	1
`beta`	0.01
`max_steps`	500
`max_prompt_length`	2048
`max_completion_length`	8192
`mask_truncated_completions`	`True`
`num_generations`	16
`temperature`	1.0
`scale_rewards`	`True`

E.3 Experiments Compute Resources

All experiments were conducted on a single server node equipped with 8 `NVIDIA A100 80GB` GPUs, 2 `Intel Xeon Platinum 8350C` CPU, and `1600GB` memory. We allocated 4 GPUs for training and 4 GPUs for *VLLM* inference. The training time for `GRPO@SATSP-Math`, `GRPO@MaxSAT-Math`, and `GRPO@SATSP-Story` was approximately 30 hours, 26 hours, and 9 hours, respectively.

[8] https://github.com/huggingface/open-r1/blob/main/src/open_r1/rewards.py

F Additional Figures

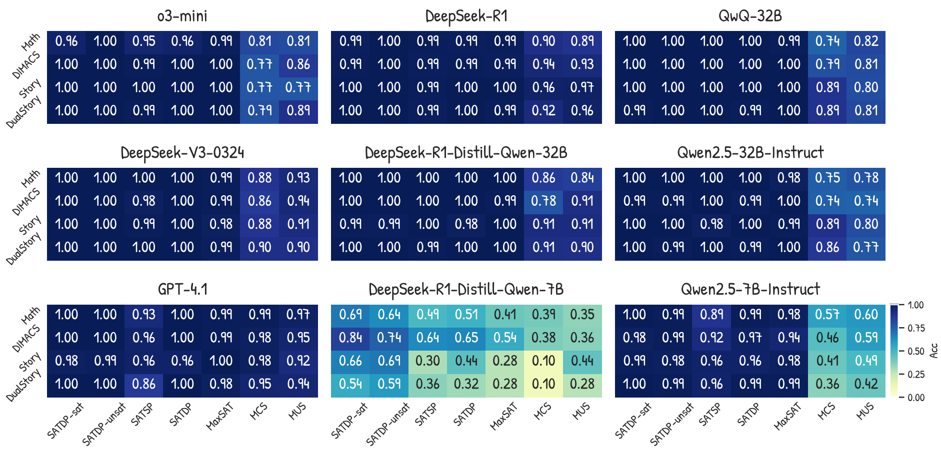

Figure 7: Format accuracy heatmaps showing LLM performance breakdown by *problem type* (columns) and *question format* (rows).

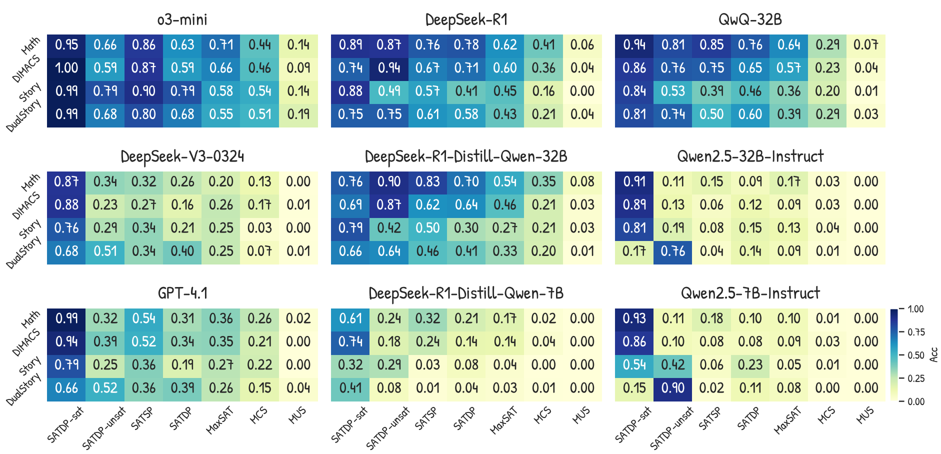

Figure 8: All accuracy heatmaps (including `SATDP-sat` and `SATDP-unsat`) showing LLM performance breakdown by *problem type* (columns) and *question format* (rows).

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: Our claims are accurately presented in the final part of Sec. 1.

   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: We discuss limitations of our work in Sec. 6.

   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: The work is empirical; no new theorems or formal proofs are introduced.

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 all necessary details about our experimental setup, datasets, and evaluation methods in Sec. 3 and 4, with additional implementation details in the Appendix.

   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: Our code and data are available at https://github.com/sdpkjc/satquest with documentation for reproducing experimental results.

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 provide comprehensive details of all experimental settings in Apps. A, B, D and E, including model configurations, hyperparameters, and evaluation protocols.

   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: [No]

   Justification: We report aggregate accuracies across instances but did not include confidence intervals or significance tests due to the large number of binary evaluation tasks.

   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 provide detailed information about computational resources in App. E.3.

   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 work involves no human subjects or privacy-sensitive data and adheres to responsible research practices per the NeurIPS Ethics Guidelines.

   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: [NA]

    Justification: Our work focuses on abstract logical reasoning evaluation methods with no direct societal impacts.

    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 released assets (benchmarks and code) pose minimal risk of misuse and require no special safeguards.

    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: All external assets (code libraries, datasets, models) used in our work are properly cited with appropriate references.

    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: [Yes]

Justification: We provide comprehensive documentation for our code and datasets in our repository README and supplementary materials.

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: Our work does not involve crowdsourcing or human-subject experiments.

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: No human subjects research was conducted.

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: We provide detailed LLM API parameters and prompt specifications in App. D. We used the Weave tool [9] to record all our evaluation results, which will be made public after the paper is accepted.

    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.

[9] https://github.com/wandb/weave