PhysProver: Advancing Automatic Theorem Proving for Physics

Anonymous ACL submission

Abstract

The combination of verifiable languages and LLMs has significantly influenced both the mathematical and computer science communities because it provides a rigorous foundation for theorem proving. Recent advancements in the field provide foundation models and sophisticated agentic systems pushing the boundaries of formal mathematical reasoning to approach the natural language capability of LLMs (Chen et al., 2025b). However, little attention has been given to the formal physics reasoning, which also heavily relies on similar problem-solving and theorem-proving frameworks. To solve this problem, this paper presents, to the best of our knowledge, the first approach to enhance formal theorem proving in the physics domain. We compose a dedicated dataset **PhysLean-Data** for the task. It is composed of theorems sampled from PhysLean (Tooby-Smith, 2025) and data generated by a conjecture-based formal data generation pipeline. To train our model, we leverage an open-source state-of-the-art mathematical theorem prover and apply Reinforcement Learning with Verifiable Rewards (RLVR) to train **PhysProver**. Comprehensive experiments demonstrate that, using only 5,000 training samples, **PhysProver** achieves a consistent **2.4%** improvement across multiple sub-domains, including difficult Quantum Field Theory problems. Furthermore, after formal physics training, we observed **1%** gains on the MiniF2F-Test benchmark, which indicates Physics domain training can, on the other hand, enhance formal math capability. The results highlight the efficiency and efficacy of our approach. To foster further research, we will release both our dataset and model to the community.

1 Introduction

Formal reasoning has long been recognized as a cornerstone of human intelligence and a critical domain in machine learning research (Newell and Simon, 1956). With the recent advancements in Large Language Models (LLMs), much research has explored their application in formal theorem proving. They explored domains from training foundation models (Lin et al., 2025b; Ren et al., 2025; Wang et al., 2025c) and specialized agent framework (Wang et al., 2025d; Chen et al., 2025b; Varambally et al., 2025). Among these, math theorem proving in Lean4 (Moura and Ullrich, 2021a) has emerged as one of the most extensively studied areas (Wang et al., 2024; Lin et al., 2025a; Xin et al., 2024). Researchers typically start from a general-purpose LLM, employing Supervised Fine-Tuning (SFT) and Reinforcement Learning (RL) to enhance the formal reasoning capability. This approach has achieved strong results on formal math benchmarks, such as MiniF2F (Zheng et al., 2022) and PutnamBench (Tsoukalas et al., 2024).

Previous works have demonstrated that developing expert models for Lean4 theorem proving demands substantial training data and a large amount of GPU hours. For instance, DeepSeek-Prover (Xin et al., 2024) applies a 120B math-related tokens continue pretraining and 8M formal statements with proofs to train an expert prover. Similarly, Goedel-Prover (Lin et al., 2025a) applies expert iteration on more than 1 million formal statements. Despite these advancements, formal theorem proving faces significant challenges due to a scarcity of high-quality data that is able to give the model a general formal reasoning capability, rather than focusing on a narrow field (Li et al., 2025).

While significant progress has been made in mathematical theorem proving, the formal physics domain remains largely overlooked. Physics, with its reliance on rigorous mathematical foundations and formal derivations, offers a natural yet underexplored extension to formal reasoning. Li et al. (2025) highlights that SOTA theorem-proving models perform poorly in physics-related tasks but fail to propose methods for improvement.

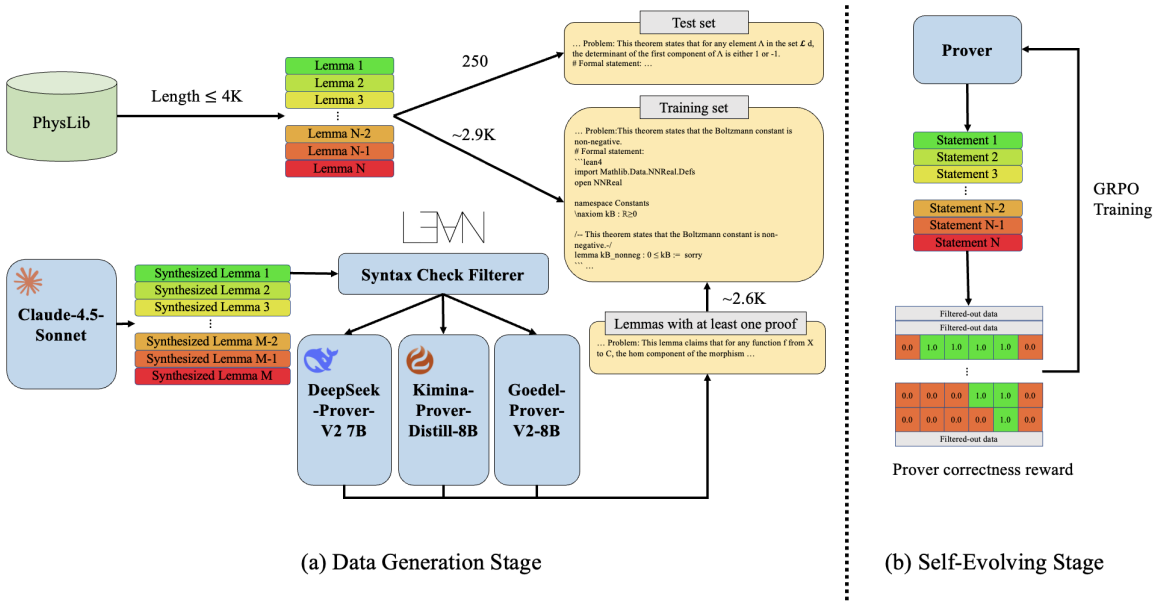

Figure 1: **Physics Prover Framework**: (a) Data Generation Stage: the training set comprises 5,541 physics statements from both PhysLib (Li et al., 2025) and synthetic lemmas from Claude-4.5-Sonnet, where the latter are further filtered by Lean syntax and proof existence checks. (b) Self-Evolving Stage: after obtaining the training set, GRPO (Shao et al., 2024) is adopted to train the base prover models, with reward signals of proof correctness provided by Lean.

To settle this gap, we, as far as we are concerned, take the first step toward enhancing theorem proving in the physics domain by constructing a specialized data pipeline and employing Reinforcement Learning with Verifiable Rewards (RLVR). The overview of our framework can be found in Figure 1. Specifically, we collect foundational theorems and lemmas from the open-source repository of PhysLean (Tooby-Smith, 2025), which contains Lean4-based results across advanced physics domains such as Quantum Field Theory and String Theory. The extracted data, along with their headers, are divided into the training and testing sets. To augment the training dataset, we apply Claude-4.5 to generate additional conjectures based on the dataset. Subsequently, we apply formal LLMs to annotate these conjectures, thereby formulating the Basic Physics Lean training dataset, which contains approximately 5,000 training samples and 250 testing samples.

With the dataset, we leverage RLVR (Lambert et al., 2025) to enhance physical theorem-proving capability using the GRPO algorithm. Our evaluation demonstrates consistent improvement across multiple physics domains and achieves a **2.4%** of overall improvements compared to SOTA math provers on the testing dataset. Furthermore, when tested on the Out-of-Distribution (OOD) MiniF2F benchmark (Zheng et al., 2022), **PhysProver** achieved over 1% of improvement compared to the base model under pass@16. It demonstrates the efficiency of our approach and shows that physics dataset training can enhance the formal math capability of the model.

We summarize our contribution as follows:

1. Introducing the first methods specifically designed to train the formal theorem provers for physics.
2. Developing and releasing a relatively small, but comprehensive dataset and a conjecture generation pipeline for physical theorems to benefit the research community.
3. Training a formal physics prover that outperforms the SOTA model and achieves superior performance in both physics and mathematical theorem proving.

2 Related Works

2.1 Formal Math Reasoning

Formal math reasoning involves representing mathematical components in a computer-verifiable format. It reduces the ambiguity and establishes a rigorous foundation for logical reasoning. Over the past decades, researchers have developed numerous Formal Languages (FLs) based on two primary theoretical frameworks. The first relies on

dependent type languages, such as Lean (De Moura et al., 2015; Moura and Ullrich, 2021b) and Coq (Coq, 1996), where formal verification is achieved through a small kernel to perform type checking. The second line utilizes higher-order logic to quantify functions and predicates. This line of work is represented by languages such as Isabelle (Paulson, 1994), HOL, and HOL Light (Harrison, 2009). Among the above languages, Lean4 (Moura and Ullrich, 2021b) has gained significant attention due to its expressiveness and extensive Mathlib4 repository, which encompasses almost all major mathematical domains.

The rise of LLMs has accelerated the advancements in formal proving tasks. Researchers have compiled extensive datasets of mathematical theorems and proofs(Wang et al., 2025c; Lin et al., 2025a; Dong and Ma, 2025), which provide a robust foundation for model training. Building on these resources, increasingly sophisticated models have emerged. Early efforts, such as Expert Iteration (Polu et al., 2022), employed iterative annotation using LLMs to enhance the training data. Open-source frameworks like DeepSeek-Prover (Xin et al., 2024) and Theorem-Llama (Wang et al., 2024) further advanced the formal provers. More recently, RLVR has enabled Long CoT training for formal theorem proving, which works like MA-LoT (Wang et al., 2025c), Kimina-Prover (Wang et al., 2025a), DeepSeek-Prover-V2 (Ren et al., 2025), and Goedel-Prover-V2 (Lin et al., 2025b), achieving notable progress. The emergence of agentic frameworks, such as Hilbert (Varambally et al., 2025) and Seed-Prover-V1 (Chen et al., 2025c), achieves notable progress by enabling multi-agent theorem decomposition and sub-goal proofs. The latest works apply agentic RL to push LLMs' formal reasoning capability closer to natural language proficiency (Chen et al., 2025b). Despite these advancements, formal reasoning in physics remains an underexplored domain, representing a significant opportunity for future research.

2.2 LLM for Physics Reasoning

With the rapid development of general reasoning capabilities in LLMs, researchers are actively exploring the application of these models in more diverse fields (Wang et al., 2025b). Among them, physics reasoning is one key field that receives significant attention. In the context of benchmarks, early comprehensive benchmarks, such as SciBench (Wang et al., 2023) and GPQA (Rein et al., 2024), evaluate college-level scientific problem-solving across multiple scientific fields, including physics. More recently, physics benchmarks have emerged at multiple difficulty levels: UGPhysics (Xu et al., 2025) presents 5,520 undergraduate-level bilingual problems that advanced thinking models are hard to solve; OlympiadBench (He et al., 2024) introduces 8,476 Olympiad-level problems with multi-module inputs; and recent HiPhO (Yu et al., 2025) compiles the latest 13 Physics Olympiad exams in 2024-2025 with human-aligned evaluation.

On the model training side, researchers began exploring the potential for LLMs as physics reasoning tools from an early stage. Early works have demonstrated that LLMs can solve complex word problems that require calculation and inference (Ding et al., 2023). Such capability can be further enhanced by Reinforcement Learning from Human Feedback (RLHF) (Anand et al., 2024) or simple multi-agent collaboration (Pang et al., 2025). Recent works apply RLVR on natural language physics problems, with P1 (Chen et al., 2025a) achieving gold-level IPhO performance. However, with a lack of datasets and training methods, developing LLMs for formal physics reasoning is relatively understudied currently (Li et al., 2025).

3 Methodology

3.1 Seed Dataset Collection

We construct a lemma–proof dataset from the PhysLean GitHub repository (Tooby-Smith, 2025) by extracting all provable lemmas from `.lean` files along with their preceding formal headers. The lemma statements with context serve as inputs, while the corresponding proof scripts serve as outputs. We filter the samples to retain only those with a total length under 4,096 tokens. The resulting corpus contains over 3,000 examples, which are randomly split into training and test sets at approximately a 9:1 ratio, yielding 2,933 training and 250 test instances. The dataset spans a broad range of domains in physics and mathematics, encompassing classical and modern physics (e.g., classical mechanics, electromagnetism, quantum mechanics, and relativity) as well as advanced theoretical areas such as quantum field theory, string theory, and mathematical foundations.

3.2 Synthetic Data Generation

To augment our dataset, we construct a conjecture generation and verification pipeline inspired by STP (Dong and Ma, 2025). Specifically, we treat our initial data as seed data, denoted as $D_{\text{seed}} = \{(h_i, l_i, p_i)\}_{i=1}^{N}$, where h_i, l_i, and p_i are the header, lemma, and corresponding proof of the i^{th} sample, and N is the total number of seed examples. For each sample, we use Claude-4.5-Sonnet (Anthropic, 2025) to generate 10 conjectures by providing the header–lemma pairs (h_i, l_i), yielding 29,330 candidate statements. The prompting template is provided in Figure 4 in the Appendix.

After collecting the conjectures, we apply a two-stage pipeline to select well-formed and correct statements. We first examine the **validity** of each conjecture. Specifically, for each conjecture c_{ij}, we append it to the corresponding header h_i to form $D_c = \{(h_i, c_{ij}) \mid i = 1, \ldots, N,\ j = 1, \ldots, 10\}$ and use the Lean verifier to check whether the statement is well-formed. This includes verifying that all variables are properly defined and that all referenced definitions and theorems exist. After this step, 6,971 conjectures remain, corresponding to a retention rate of 23.8%.

The second stage examines the **correctness** of conjectures. Given a conjecture with its corresponding header, we leverage DeepSeek-Prover-V2-7B (Ren et al., 2025), Goedel-Prover-V2-8B (Lin et al., 2025b), and Kimina-Prover-Distill-8B (Wang et al., 2025a) to generate 16 proofs, producing response samples $\{(h_i, c_{ij}, r_p)\}_{p=1}^{16}$. A conjecture is deemed correct if

$$\exists p,\ 1 \le p \le 16 : \texttt{Verify}(h_i, c_{ij}, r_p) = \texttt{True}$$

where `Verify` denotes the Lean verification result. This process yields 2,608 verified conjectures, representing an overall pipeline yield rate of 8.9%, comparable to STP (Dong and Ma, 2025). Combining these with the 2,933 seed training examples produces a total of 5,541 training instances for our experiments.

Notably, we also compared different proprietary models, including GPT-5 (OpenAI, 2025) and Gemini-2.5-Pro (Google, 2025). However, the validity rates of their generated conjectures were substantially lower than those produced by Claude. We additionally explored generating conjectures in natural language and converting them to Lean4 statements using an auto-formalizer. However, the dependencies in physical statements are complex, making it difficult to identify a uniform header for the auto-formalizer. Consequently, this approach also yielded low success rates.

3.3 Self-Evolving Pipeline

We conduct Reinforcement Learning (RL) to lift the performance on physics domain. Specifically, our experiments are mainly based on Group Relative Policy Optimization (GRPO) (Shao et al., 2024). For each prompt x in the training set, they sample G (Group size) responses during the rollout stage, and optimize the following objective:

$$\begin{aligned}\mathcal{J}_{\text{GRPO}}(\theta) = \mathbb{E}_{x\sim\mathcal{D},\{y_i\}_{i=1}^{G}\sim\pi_{\theta_{\text{old}}}(\cdot|x)} \\ \left[\frac{1}{G}\sum_{i=1}^{G}\frac{1}{|y_i|}\sum_{t=1}^{|y_i|}\min\Big(w_{i,t}(\theta)\hat{A}_{i,t},\right. \\ \left.\text{clip}\left(w_{i,t}(\theta), 1-\varepsilon, 1+\varepsilon\right)\hat{A}_{i,t}\Big)\right] \\ -\beta\,\mathbb{D}_{\text{KL}}\left(\pi_\theta \| \pi_{\theta_{\text{ref}}}\right)\end{aligned}$$

where y_i is the i^{th} generated sequence of tokens, ε is the clip ratio. The importance ratio $w_{i,t}(\theta)$ and the advantage $\hat{A}_{i,t}$ are calculated as follows:

$$w_{i,t}(\theta) = \frac{\pi_\theta(y_{i,t}|x, y_{i,<t})}{\pi_{\theta_{\text{old}}}(y_{i,t}|x, y_{i,<t})},$$

$$\hat{A}_{i,t} = \hat{A}_i = \frac{r(x, y_i) - \text{mean}\left(\{r(x, y_i)\}_{i=1}^{G}\right)}{\text{std}\left(\{r(x, y_i)\}_{i=1}^{G}\right)},$$

respectively. And all the tokens in y_i share the same advantage as $\hat{A}_{i,t}$.

4 Experiments

To evaluate our methodology, we use **PhysLeanData** to fine-tune popular Lean-based formal mathematics provers. Our experiments reveal that strong mathematical reasoning models exhibit notable limitations when handling formal physics problems, underscoring the importance of domain-specific formal datasets and self-improving data augmentation strategies.

4.1 Experimental Setup

4.1.1 Dataset and Tasks

Model performance is evaluated on the test set of **PhysLeanData**, which shares the same source as the training set with a 9:1 train-test split. To ensure fair comparison across models with different context lengths, we retain only samples with prompt

Method	Budget	Classical	Particle & String	Relativity	Quantum Field Theory	Overall
Proprietary Models						
GPT-5 (OpenAI, 2025)	pass@16	37.3	13.4	21.3	35.2	26.4
Claude-4.5-Sonnet (Anthropic, 2025)	pass@16	52.9	19.4	29.5	**39.4**	34.4
Formal Math Provers						
Kimina-Prover-Distill-8B (Wang et al., 2025a)	pass@16	35.3	14.9	29.5	22.5	24.8
Goedel-Prover-V2-8B (Lin et al., 2025b)	pass@16	49.0	19.4	34.4	28.2	31.6
Deepseek-Prover-V2-7B (Ren et al., 2025)	pass@16	54.9	23.9	37.7	25.4	34.0
Formal Physics Provers						
Deepseek-Prover-V2 + PhysLeanData	pass@16	**58.8 (+3.9)**	**26.9 (+3.0)**	**39.3 (+1.6)**	26.8 (+1.4)	**36.4 (+2.4)**

Table 1: **Main experimental results**. We evaluate all the models on **PhysLeanData** test set, which includes Classical, Particle & String, Relativity, and Quantum Field Theory domains. We report the pass@16 accuracy.

lengths under 4,096 tokens, resulting in 250 lemmas in the final evaluation set.

For finer-grained analysis, we organize the test samples into four physics categories: Classical & Foundational Physics, Particle & String Physics, Relativity & Spacetime, and Quantum Field Theory. This classification reflects distinct theoretical frameworks and varying levels of required domain expertise. Further details are provided in Appendix A.

4.1.2 Models and Baselines

We compare several popular open-source prover models, including DeepSeek-Prover-V2-7B (Ren et al., 2025), Goedel-Prover-V2-8B (Lin et al., 2025b), and Kimina-Prover-Distill-8B (Wang et al., 2025a), all of which are strong formal theorem provers tailored for mathematical domains. Among these, DeepSeek-Prover-V2-7B performs slightly better than the others. Therefore, our experiments focus on fine-tuning the DeepSeek prover to push the boundaries of open-source models.

For baselines, we first report the performance of DeepSeek-Prover-V2-7B, Kimina-Prover-Distill-8B, and Goedel-Prover-V2-8B without any additional training. We also include comparisons with strong proprietary systems, namely GPT-5 (OpenAI, 2025) and Claude-4.5-Sonnet (Anthropic, 2025). For all baselines, we use a fixed sampling budget and report pass@16 accuracy, ensuring fair comparison under a consistent inference budget. For open-source provers, we use the prompt template provided in Appendix B.1. For proprietary models, we employ a tailored Chain-of-Thought (CoT) (Wei et al., 2023) prompt to encourage step-by-step reasoning before generating the final proof.

4.2 Implementation Details

We directly apply Reinforcement Learning starting from the DeepSeek-Prover-V2-7B using verl (Sheng et al., 2025). Specifically, we apply GRPO with rule-based rewards (Lambert et al., 2025; DeepSeek-AI et al., 2025). We integrate the Lean verifier into the verl framework and use it to judge the proofs. The version of Lean Verifier we are using is `4.20.0`. The reward score for each trajectory is calculated as follows:

$$r(x, y_i) = \begin{cases} 1 & \text{if Verify}(x, y_i) = \text{True} \\ 0 & \text{otherwise} \end{cases}$$

Additionally, if the proof contains `sorry`, `admit`, or `apply?` keywords, we directly assign a 0 for the reward score. To allow a smooth transition of difficulties during the learning process, curriculum learning (Parashar et al., 2025) is employed by sorting the lemma based on their groundtruth proving length.

We train all models on 8×H200 GPUs with a constant learning rate of $1e^{-6}$ and a batch size of 256 for 2 epochs, and the training takes approximately 8 hours. Notably, we do not begin with Supervised Fine-Tuning (SFT) because we observe that it degrades performance; we analyze this behavior in Section 6.

4.3 Experiment Results

Our experimental results are presented in Table 1. We first observe that all existing models achieve relatively low scores despite their proficiency in mathematical theorem proving, with none exceeding 40% accuracy. Notably, open-source theorem provers exhibit competitive accuracy compared to the latest proprietary systems, such as Claude-4.5-Sonnet and GPT-5. However, proprietary models demonstrate different strengths across physical domains compared to their open-source counterparts. For instance, all open-source provers achieve below 30% accuracy on Quantum Field Theory, whereas proprietary models exceed 35%. This suggests that

Header and Context Lemmas

```
import PhysLean.QFT.PerturbationTheory.WickAlgebra.
      NormalOrder.Lemmas
import PhysLean.QFT.PerturbationTheory.WickAlgebra.
      TimeOrder

namespace FieldSpecification
variable {𝓕 : FieldSpecification}

def timeContract (φ ψ: 𝓕.FieldOp) : 𝓕.WickAlgebra
      :=
    𝓣(ofFieldOp φ* ofFieldOp ψ) - 𝓣(ofFieldOp φ*
         ofFieldOp ψ)

lemma timeContract_eq_superCommute (φ ψ: 𝓕.FieldOp)
      :
    timeContract φψ= if timeOrderRel φψthen [anPart
          φ, ofFieldOp ψ]ₛ
    else 𝓢(𝓕 |>ₛ φ, 𝓕|>ₛ ψ) ·[anPart ψ, ofFieldOp
          φ]ₛ := by
  split_ifs
  ·rename_i h
    rw [timeContract_of_timeOrderRel _ _ h]
  ·rename_i h
    rw [timeContract_of_not_timeOrderRel_expand _ _ h
          ]

lemma timeContract_zero_of_diff_grade (φ ψ: 𝓕.
      FieldOp) (h : (𝓕 |>ₛ φ) ≠(𝓕 |>ₛ ψ)) :
    timeContract φψ= 0 := by
  by_cases h1 : timeOrderRel φψ
  ·rw [timeContract_of_timeOrderRel _ _ h1]
    rw [superCommute_anPart_ofFieldOpF_diff_grade_zero
          ]
    exact h
  ·rw [timeContract_of_not_timeOrderRel _ _ h1]
    rw [timeContract_of_timeOrderRel _ _ _]
    rw [superCommute_anPart_ofFieldOpF_diff_grade_zero
          ]
    simp only [instCommGroup.eq_1, sμl_zero]
    exact h.symm
    have ht := IsTotal.total (r := 𝓕.timeOrderRel) φ
          ψ
    simp_all
    ·····
```

DeepseekProver + PhysProver

```
/-- This lemma claims that the normal ordering of the
      time-contracted product of any two field
     operators φand ψis identically zero.-/
lemma normalOrder_timeContract (φ ψ: 𝓕.FieldOp) :
    𝓝(timeContract φψ) = 0 := by
  rw [timeContract_eq_superCommute]
  split_ifs <;> simp_all [timeContract, superCommu
       te_anPart_ofFieldOpF_diff_grade_zero,
    sμl_eq_zero, map_zero]
```

DeepseekProver

```
lemma normalOrder_timeContract (φ ψ: 𝓕.FieldOp) :
    𝓝(timeContract φψ) = 0 := by
  rw [timeContract_eq_superCommute]
  split_ifs
  ·rw [normalOrder_ofFieldOp_pair_eq_zero]
    <;> simp_all [timeOrderRel_of_isContraction]
  ·rw [normalOrder_anPart_ofFieldOp_sμl]
    <;> simp_all [timeOrderRel_of_isContraction]
```

Figure 2: Successful from the **PhysProver** and failed proofs from the base model for the same statements. **PhysProver** demonstrates better in-context learning ability to make good usage of lemmas.

proprietary and open-source models may be trained on different mixtures of physics data.

Our trained model, FormalPhysics, substantially surpasses its formal mathematics prover counterparts, consistently achieving gains across all categories. Specifically, on the most challenging domains—Quantum Field Theory and Particle & String Physics—where all baselines exhibit low accuracy, our model still yields notable improvements. These results demonstrate the effectiveness of training a mathematics prover for the physics domain using only approximately 5K samples. On top of that, FormalPhysics, with only 7B parameters, outperforms Claude-4.5-Sonnet on formal physics theorem proving.

5 Analysis

5.1 Improved In-Context Learning Through Reinforcement Learning

In this subsection, we provide a detailed analysis of the performance gains achieved by **PhysProver** through a comparative examination of proofs generated by the baseline model and **PhysProver**. Figure 2 presents an illustrative example from our test set along with the corresponding generations. The header and lemmas constitute the context for physical theorem proving, where the lemmas may serve as auxiliary tools during the proof process. The second box displays the proof completion from **PhysProver**, while the third box shows the completion from the base model, DeepSeek-Prover-V2-7B. We observe that **PhysProver** consistently makes correct use of functions and lemmas, with successful applications highlighted in blue. For instance, to prove the given lemma, **PhysProver** first applies *timeContract_eq_superCommute*, followed by the function *timeContract*. Subsequently, the model correctly invokes *superCommute_anPart_ofFieldOpF_diff_grade_zero*, demonstrating effective utilization of contextual information. By synthesizing the knowledge provided in the context, **PhysProver** successfully completes the proof. In contrast, while the base model initially applies *timeContract_eq_superCommute* correctly, it subsequently generates hallucinated content, including non-existent lemmas such as *normalOrder_ofFieldOp_pair_eq_zero* and *timeOrderRel_of_isContraction* (marked in red). These observations suggest that the reinforcement learning process on **PhysLeanData** enhances performance by enabling the model to better

leverage contextual information and comprehend domain-specific terminology. This finding also accounts for the low accuracy observed across all base models: their unfamiliarity with physics-specific lemmas and contextual structures impedes their ability to effectively utilize these resources for proof completion.

5.2 Out-of-Distribution Generalization

Surprisingly, we also observe that training on physics-centered problems yields notable generalization improvements in formal mathematical theorem proving. In this subsection, we evaluate our trained model on MiniF2F-Test (Zheng et al., 2022), which comprises 244 Lean4 statements in the mathematics domain, ranging from high school competition problems to elementary undergraduate-level proofs. We partition the dataset into several categories following Ren et al. (2025). For each statement in MiniF2F-Test, we prompt both the baseline and our trained model to generate 16 trajectories and compute pass@16 accuracy. We use the same prompt template from the DeepSeek website [1].

As shown in Table 2, models trained on **PhysLeanData** overall outperform their base versions. Specifically, our GRPO model solves 3 additional problems from the test set. However, the improvement is not consistent across all categories. For example, our model demonstrates meaningful gains on medium-level problems from MATH (Hendrycks et al., 2021). Conversely, more challenging Olympiad-level problems may not benefit from GRPO training, as performance drops in the AIME category.

These results reveal both the intrinsic connections and distinctions between mathematical and physical theorem proving in Lean4. In general, training on physics problems can enhance mathematical reasoning capabilities. However, difficult mathematics problems may demand substantially different problem-solving skills that cannot be directly acquired from physics-based training.

6 Revisiting the Role of Supervised Fine-tuning

We additionally investigated whether conducting Supervised Fine-tuning (SFT) prior to Reinforcement Learning on **PhysLeanData** could enhance model performance on Physics, following standard practice in training specialized LLMs. However, we did not observe any improvement on our test set after SFT; instead, we observed consistent performance degradation. Specifically, we fine-tuned on the **PhysLeanData** training set, where ground-truth answers were either extracted from the PhysLean library or generated by open-source provers with subsequent verification. The training sample template follows the RL prompt template in B.1, with loss computation restricted to the completion portion. We fine-tuned Deepseek-Prover-V2-7B for one epoch, using a learning rate of $5e^{-7}$ and a batch size of 32. Results are presented in Table 3, revealing consistent performance degradation across all categories, with an average accuracy decline of 6.4To gain preliminary insight into this phenomenon, we conducted experiments comparing the uncertainty of the SFT model (Table 3) and the GRPO model from our main experiments, hereafter referred to as DS-Prover-SFT and DS-Prover-GRPO, respectively. To assess model uncertainty on both training and test data, we measured the average perplexity of sampled responses conditioned on input prompts. Given a prompt x x from either the training or test set, we sampled $K = 16$ responses y_k from the model and computed the mean perplexity across these samples. We randomly selected 50 samples from each of the training and test sets. The computation is defined as:

$$\overline{PPL}(x) = \frac{1}{K}\sum_{k=1}^{K} PPL(y^{(k)}), \quad y^{(k)} \sim p_\theta(\cdot \mid x)$$

where

$$PPL(y) = \exp\left(-\frac{1}{|y|}\sum_{t=1}^{|y|}\log p_\theta(y_t \mid y_{<t}, x)\right).$$

This metric captures the model's self-uncertainty: lower values indicate that the model generates responses it considers likely and more familiar with the input, while higher values suggest greater variability or unfamiliarity with the prompt.

Results are presented in Table 4. The findings demonstrate that the average perplexity for DS-Prover-GRPO is substantially lower than that of DS-Prover-SFT on both the training and test sets, which explains why GRPO improves the performance while SFT does not. These results suggest that although supervised fine-tuning directly maximizes the probability of target tokens, it does not

[1] https://huggingface.co/deepseek-ai/DeepSeek-Prover-V2-7B

Problem Category		**Deepseek-Prover-V2** Pass@16	**Deepseek-Prover-V2 + PhysLeanData** Pass@16
Olympiad	IMO	4/20 = 20.0%	4/20 = 20.0%
	AIME	**8/15 = 53.3%**	7/15 = 46.7%
	AMC	25/45 = 55.6%	25/45 = 55.6%
MATH	Algebra	63/70 = 90.0%	**65/70 = 92.9%**
	Number Theory	51/60 = 85.0%	**53/60 = 88.3%**
Custom	Algebra	8/18 = 44.4%	8/18 = 44.4%
	Number Theory	4/8 = 50.0%	4/8 = 50.0%
	Induction	4/8 = 50.0%	4/8 = 50.0%
Overall Pass Rate		167/244 = 68.4%	**170/244 = 69.7%**

Table 2: **Out-of-Distribution Generalization** in Formal Math Proving on MiniF2F-Test (Zheng et al., 2022)

Method	**Budget**	**Classical**	**Particle & String**	**Relativity**	**Quantum Field Theory**	**Overall**
Deepseek-Prover-V2-7B	pass@16	54.9	23.9	37.7	25.4	34.0
Deepseek-Prover-V2-7B + Phys SFT	pass@16	45.1 (-9.8)	19.4 (-4.5)	26.2 (-11.5)	23.9 (-1.5)	27.6 (-6.4)

Table 3: Experiment Results of Supervised Fine-tuning (SFT) on **PhysLeanData** of Deepseek-Prover-V2-7B. The pass@16 accuracy drops significantly after this stage.

necessarily reduce model uncertainty, particularly for models such as DeepSeek-Prover that have already undergone extensive domain-specific training. This observation offers an important insight for further improving expert models: supervised fine-tuning may not always be necessary or optimal. Direct application of reinforcement learning can serve as a viable alternative, particularly in low-resource settings.

	Training Set	Test Set
DS-Prover-SFT	1.817	1.711
DS-Prover-GRPO	**1.141**	**1.209**

Table 4: The experiment results of perplexity on the training set and the test set for DS-Prover-SFT and DS-Prover-GRPO.

7 Conclusion

In this paper, we present the first systematic effort to advance formal theorem proving in the physical domain. We first introduce **PhysLeanData**, a dataset of physical theorems formalized in Lean4, along with a conjecture formulation pipeline for generating valid and correct conjectures. By applying Reinforcement Learning with Verifiable Rewards (RLVR) to an open-source state-of-the-art theorem prover, our **PhysProver** achieves consistent **2.4%** improvements across physical subdomains such as Quantum Field Theory using only 5K samples. The model also demonstrates over **1%** improvement on the out-of-distribution MiniF2F-test benchmark, highlighting strong generalization capability. Our work bridges a critical gap between formal theorem proving in mathematics and its application to the physical sciences. We will publicly release our dataset and models to facilitate future research in this direction.

8 Limitations

Our work has several limitations that we acknowledge and hope to address in future research. First, due to computational resource constraints, we were unable to collect more data or scale the conjecture generation process to a larger extent. As noted in Section 3.2, our synthetic data pipeline has a yield rate of only 8.9%, meaning that a substantial portion of generated conjectures are filtered out during validity and correctness verification. Scaling up the generation process would require significantly more compute for both the LLM-based conjecture generation and the multi-prover verification stage, which was beyond our current budget. Additionally, our dataset is derived solely from the PhysLean repository, which, while comprehensive, may not cover all areas of physics uniformly. Certain specialized domains may be underrepresented, potentially limiting the model's applicability to the full breadth of physical theorem proving.

References

Avinash Anand, Kritarth Prasad, Chhavi Kirtani, Ashwin R Nair, Mohit Gupta, Saloni Garg, Anurag Gautam, Snehal Buldeo, and Rajiv Ratn Shah. 2024. Enhancing llms for physics problem-solving using reinforcement learning with human-ai feedback. *arXiv preprint arXiv:2412.06827*.

Anthropic. 2025. Claude sonnet 4.5 system card. Technical report, Anthropic.

Jiacheng Chen, Qianjia Cheng, Fangchen Yu, Haiyuan Wan, Yuchen Zhang, Shenghe Zheng, Junchi Yao, Qingyang Zhang, Haonan He, Yun Luo, and 1 others. 2025a. P1: Mastering physics olympiads with reinforcement learning. *arXiv preprint arXiv:2511.13612*.

Jiangjie Chen, Wenxiang Chen, Jiacheng Du, Jinyi Hu, Zhicheng Jiang, Allan Jie, Xiaoran Jin, Xing Jin, Chenggang Li, Wenlei Shi, and 1 others. 2025b. Seed-prover 1.5: Mastering undergraduate-level theorem proving via learning from experience. *arXiv preprint arXiv:2512.17260*.

Luoxin Chen, Jinming Gu, Liankai Huang, Wenhao Huang, Zhicheng Jiang, Allan Jie, Xiaoran Jin, Xing Jin, Chenggang Li, Kaijing Ma, and 1 others. 2025c. Seed-prover: Deep and broad reasoning for automated theorem proving. *arXiv preprint arXiv:2507.23726*.

Projet Coq. 1996. The coq proof assistant-reference manual. *INRIA Rocquencourt and ENS Lyon, version*, 5.

Leonardo De Moura, Soonho Kong, Jeremy Avigad, Floris Van Doorn, and Jakob von Raumer. 2015. The lean theorem prover (system description). In *Automated Deduction-CADE-25: 25th International Conference on Automated Deduction, Berlin, Germany, August 1-7, 2015, Proceedings 25*, pages 378–388. Springer.

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, and 181 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *Preprint*, arXiv:2501.12948.

Jingzhe Ding, Yan Cen, and Xinyuan Wei. 2023. Using large language model to solve and explain physics word problems approaching human level. *arXiv preprint arXiv:2309.08182*.

Kefan Dong and Tengyu Ma. 2025. Stp: Self-play llm theorem provers with iterative conjecturing and proving. *Preprint*, arXiv:2502.00212.

Google. 2025. Gemini 2.5: Our newest gemini model with thinking. https://blog.google/technology/google-deepmind/gemini-model-thinking-updates-march-2025/. Accessed: 2025.

John Harrison. 2009. Hol light: An overview. In *International Conference on Theorem Proving in Higher Order Logics*, pages 60–66. Springer.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, and 1 others. 2024. Olympiadbench: A challenging benchmark for promoting agi with olympiad-level bilingual multimodal scientific problems. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3828–3850.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring mathematical problem solving with the math dataset. *Preprint*, arXiv:2103.03874.

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, and 4 others. 2025. Tulu 3: Pushing frontiers in open language model post-training. *Preprint*, arXiv:2411.15124.

Yuxin Li, Minghao Liu, Ruida Wang, Wenzhao Ji, Zhitao He, Rui Pan, Junming Huang, Tong Zhang, and Yi R Fung. 2025. Lean4physics: Comprehensive reasoning framework for college-level physics in lean4. *arXiv preprint arXiv:2510.26094*.

Yong Lin, Shange Tang, Bohan Lyu, Jiayun Wu, Hongzhou Lin, Kaiyu Yang, Jia Li, Mengzhou Xia, Danqi Chen, Sanjeev Arora, and Chi Jin. 2025a. Goedel-prover: A frontier model for open-source automated theorem proving. *Preprint*, arXiv:2502.07640.

Yong Lin, Shange Tang, Bohan Lyu, Ziran Yang, Jui-Hui Chung, Haoyu Zhao, Lai Jiang, Yihan Geng, Jiawei Ge, Jingruo Sun, and 1 others. 2025b. Goedel-prover-v2: Scaling formal theorem proving with scaffolded data synthesis and self-correction. *arXiv preprint arXiv:2508.03613*.

Leonardo de Moura and Sebastian Ullrich. 2021a. The lean 4 theorem prover and programming language. In *Automated Deduction – CADE 28*, pages 625–635, Cham. Springer International Publishing.

Leonardo de Moura and Sebastian Ullrich. 2021b. The lean 4 theorem prover and programming language. In *Automated Deduction–CADE 28: 28th International Conference on Automated Deduction, Virtual Event, July 12–15, 2021, Proceedings 28*, pages 625–635. Springer.

Allen Newell and Herbert Alexander Simon. 1956. *The Logic Theory Machine: A Complex Information Processing System*. RAND Corporation, Santa Monica, CA.

OpenAI. 2025. GPT-5 system card. Technical report, OpenAI.

Xinyu Pang, Ruixin Hong, Zhanke Zhou, Fangrui Lv, Xinwei Yang, Zhilong Liang, Bo Han, and Changshui Zhang. 2025. Physics reasoner: Knowledge-augmented reasoning for solving physics problems with large language models. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 11274–11289.

Shubham Parashar, Shurui Gui, Xiner Li, Hongyi Ling, Sushil Vemuri, Blake Olson, Eric Li, Yu Zhang, James Caverlee, Dileep Kalathil, and Shuiwang Ji. 2025. Curriculum reinforcement learning from easy to hard tasks improves llm reasoning. *Preprint*, arXiv:2506.06632.

Lawrence C Paulson. 1994. *Isabelle: A generic theorem prover*. Springer.

Stanislas Polu, Jesse Michael Han, Kunhao Zheng, Mantas Baksys, Igor Babuschkin, and Ilya Sutskever. 2022. Formal mathematics statement curriculum learning. *arXiv preprint arXiv:2202.01344*.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. 2024. Gpqa: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*.

ZZ Ren, Zhihong Shao, Junxiao Song, Huajian Xin, Haocheng Wang, Wanjia Zhao, Liyue Zhang, Zhe Fu, Qihao Zhu, Dejian Yang, and 1 others. 2025. Deepseek-prover-v2: Advancing formal mathematical reasoning via reinforcement learning for subgoal decomposition. *arXiv preprint arXiv:2504.21801*.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, and 1 others. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. 2025. Hybridflow: A flexible and efficient rlhf framework. In *Proceedings of the Twentieth European Conference on Computer Systems*, EuroSys '25, page 1279–1297. ACM.

Joseph Tooby-Smith. 2025. Heplean: Digitalising high energy physics. *Computer Physics Communications*, 308:109457.

George Tsoukalas, Jasper Lee, John Jennings, Jimmy Xin, Michelle Ding, Michael Jennings, Amitayush Thakur, and Swarat Chaudhuri. 2024. Putnambench: Evaluating neural theorem-provers on the putnam mathematical competition. *Preprint*, arXiv:2407.11214.

Sumanth Varambally, Thomas Voice, Yanchao Sun, Zhifeng Chen, Rose Yu, and Ke Ye. 2025. Hilbert: Recursively building formal proofs with informal reasoning. *Preprint*, arXiv:2509.22819.

Haiming Wang, Mert Unsal, Xiaohan Lin, Mantas Baksys, Junqi Liu, Marco Dos Santos, Flood Sung, Marina Vinyes, Zhenzhe Ying, Zekai Zhu, and 1 others. 2025a. Kimina-prover preview: Towards large formal reasoning models with reinforcement learning. *arXiv preprint arXiv:2504.11354*.

Ruida Wang, Yuxin Li, Tong Zhang, and 1 others. 2025b. Let's reason formally: Natural-formal hybrid reasoning enhances llm's math capability. *arXiv preprint arXiv:2505.23703*.

Ruida Wang, Rui Pan, Yuxin Li, Jipeng Zhang, Yizhen Jia, Shizhe Diao, Renjie Pi, Junjie Hu, and Tong Zhang. 2025c. Ma-lot: Model-collaboration lean-based long chain-of-thought reasoning enhances formal theorem proving. *arXiv preprint arXiv:2503.03205*.

Ruida Wang, Jiarui Yao, Rui Pan, Shizhe Diao, and Tong Zhang. 2025d. Gar: Generative adversarial reinforcement learning for formal theorem proving. *arXiv preprint arXiv:2510.11769*.

Ruida Wang, Jipeng Zhang, Yizhen Jia, Rui Pan, Shizhe Diao, Renjie Pi, and Tong Zhang. 2024. Theoremllama: Transforming general-purpose llms into lean4 experts. *Preprint*, arXiv:2407.03203.

Xiaoxuan Wang, Ziniu Hu, Pan Lu, Yanqiao Zhu, Jieyu Zhang, Satyen Subramaniam, Arjun R Loomba, Shichang Zhang, Yizhou Sun, and Wei Wang.

2023. Scibench: Evaluating college-level scientific problem-solving abilities of large language models. *arXiv preprint arXiv:2307.10635*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2023. Chain-of-thought prompting elicits reasoning in large language models. *Preprint*, arXiv:2201.11903.

Huajian Xin, Daya Guo, Zhihong Shao, Zhizhou Ren, Qihao Zhu, Bo Liu, Chong Ruan, Wenda Li, and Xiaodan Liang. 2024. Deepseek-prover: Advancing theorem proving in llms through large-scale synthetic data. *Preprint*, arXiv:2405.14333.

Xin Xu, Qiyun Xu, Tong Xiao, Tianhao Chen, Yuchen Yan, Jiaxin Zhang, Shizhe Diao, Can Yang, and Yang Wang. 2025. Ugphysics: A comprehensive benchmark for undergraduate physics reasoning with large language models. *arXiv preprint arXiv:2502.00334*.

Fangchen Yu, Haiyuan Wan, Qianjia Cheng, Yuchen Zhang, Jiacheng Chen, Fujun Han, Yulun Wu, Junchi Yao, Ruilizhen Hu, Ning Ding, and 1 others. 2025. Hipho: How far are (m) llms from humans in the latest high school physics olympiad benchmark? *arXiv preprint arXiv:2509.07894*.

Kunhao Zheng, Jesse Michael Han, and Stanislas Polu. 2022. Minif2f: a cross-system benchmark for formal olympiad-level mathematics. *Preprint*, arXiv:2109.00110.

A PhysProver Categories

The Classical & Foundational Physics category groups core undergraduate-level subjects, including mathematical methods, classical mechanics, quantum mechanics, statistical mechanics, and electromagnetism. These areas represent foundational discoveries in physics and are primarily textbook-driven, with standardized problem formulations and solution methods.

Particle and String Physics is grouped separately to capture topics centered on high-energy physics and fundamental interactions, often motivated by experimental programs such as those at the Large Hadron Collider. String theory topics are included in this category due to their close conceptual alignment with high-energy theoretical frameworks.

Quantum Field Theory and Relativity are treated as distinct categories due to their advanced mathematical structure and conceptual complexity. Both subjects are typically introduced at the graduate level, with quantum field theory extending quantum mechanics and relativity providing a foundational framework for spacetime and gravitation.

B Experimental Details

B.1 Prompt Template

We list the prompt template for DeepSeek-Prover-V2-7B in Figure B.1. The prompt template for Kimina-Prover and Goedel-Prover is exactly the same except for the special tokens. We chose this template because we found it stable for the models to generate proof completions, as they are provided with context and a statement in the <|Assistant|> round to directly generate the proof.

Figure 3: Prompt template for DeepSeek Prover

DeepSeek Prompt Template

````
<|begin_of_sentence|> You are an expert in mathematics, physics and Lean 4.
<|User|>Think about and solve the following problem step by step in Lean 4. →Directly give
    the formal statement in Lean 4.

# Problem: This theorem states that the matrix product of the Minkowski metric with itself
    yields the identity matrix.
# Formal statement:
```lean4
import Mathlib.Algebra.Lie.Classical
import Mathlib.Analysis.Normed.Ring.Lemmas

open Matrix

def minkowskiMatrix {d : ℕ} : Matrix (Fin 1 ⊕Fin d) (Fin 1 ⊕Fin d) ℝ:=
  LieAlgebra.Orthogonal.indefiniteDiagonal (Fin 1) (Fin d) ℝ

namespace minkowskiMatrix

variable {d : ℕ}

scoped[minkowskiMatrix] notation "η" => minkowskiMatrix

/-- This theorem states that the matrix product of the Minkowski metric with itself yields the
     identity matrix.-/
@[simp]
lemma sq : @minkowskiMatrix d * minkowskiMatrix = 1 :=
  sorry
```

<|Assistant|>
```lean4
import Mathlib.Algebra.Lie.Classical
import Mathlib.Analysis.Normed.Ring.Lemmas

open Matrix

def minkowskiMatrix {d : ℕ} : Matrix (Fin 1 ⊕Fin d) (Fin 1 ⊕Fin d) ℝ:=
  LieAlgebra.Orthogonal.indefiniteDiagonal (Fin 1) (Fin d) ℝ

namespace minkowskiMatrix

variable {d : ℕ}

scoped[minkowskiMatrix] notation "η" => minkowskiMatrix

/-- This theorem states that the matrix product of the Minkowski metric with itself yields the
     identity matrix.-/
@[simp]
lemma sq : @minkowskiMatrix d * minkowskiMatrix = 1 :=
````
````

Figure 4: Prompt template for Claude-4.5-Sonnet

Claude-4.5-Sonnet Prompt Template

```
You are an expert in mathematics, physics and Lean 4.
You are provided a context, a lemma, and a proof. Your task is to generate a list of 10
    related physics conjecture in formal language based on the context and the seed language
    statements.

The conjectures should be:
1. A meaningful variant of the original theorem: modify hypotheses, generalize structures, or
    extend scope while keeping the core mathematical insight.
2. Must differ significantly in mathematical content (changed assumptions, stronger/weaker
    conclusions, or different algebraic structures) but remain recognizably related.
3. The new conjecture should be in formal language.
4. Do not include the proof.

When generating the conjectures, preserve all specific Lean identifiers exactly as they appear
     in the formal statement. You can also refer to the original formal statement.

Context:
{context}

Natural Language Statement:
{nq}

Original Formal Statement:
{theorem}

Return the final conjectures in JSON format as a dictionary where:
- The key is "conjectures"
- The value is a list of dictionaries
- Each dictionary in the list has a key "statement" whose value is a string containing one
    conjecture

Please read, understand, and then generate a list of conjectures.
```