Robust Mathematical Problem Solver

Project Report

Programme Cohort	Large Language Models & Generative AI
Course Group	GEA-Group 2
Group Name	Robust Mathematical Problem Solver
Group Members	Liu Peixuan, Long Haobo, Cao Shuhan, Wu Jiayu, Wu Xuanyu, Wei Yanran, Jin Huiyan

Abstract

This document presents the project overview for the "Robust Mathematical Problem Solver" research initiative. The system incorporates adversarial robustness detection, natural language reasoning, and formal logic verification to enhance the reliability of large language models in mathematical problem-solving tasks. The approach is evaluated using adversarially perturbed datasets, demonstrating substantial improvements in correctness and reasoning consistency.

1 Project Overview

1.1 Project Title and Team

This project, undertaken by GEA-Group 2 (entitled *Robust Mathematical Problem Solver*), forms a part of the Large Language Models & Generative AI cohort. The team consists of seven members with distinct responsibilities, each contributing to specialized components spanning algorithmic innovation, systems integration, data curation, and evaluation.

1.2 Project Summary

The central objective of this project is to address a critical limitation in contemporary large language models (LLMs)—namely, their vulnerability to subtle perturbations, adversarial inputs, and inconsistencies in mathematical reasoning (Rajeev et al. n.d.). We propose a novel workflow that integrates robust anti-interference mechanisms with reasoning paradigms, where natural language processing capabilities are coupled with formal logic verification systems (Liu et al. n.d., Ren et al. n.d., Kumarappan et al. n.d.).

The system employs a multi-stage pipeline: first, an adversarial filtering unit detects and mitigates potentially confusing or malicious inputs, drawing upon recent advancements in adversarial trigger detection for reasoning models (Rajeev et al. n.d.). Subsequently, a reasoning core generates candidate solutions in natural language, which are subjected to formal verification within the Lean 4 theorem-proving environment (Patel et al. n.d., Baba et al. n.d., Chen et al. n.d.). A closed feedback loop then provides corrective feedback to the model, guiding iterative improvements and ensuring consistency. This design directly addresses the need for verification mechanisms in hallucination-prone reasoning systems (Liu et al. n.d.).

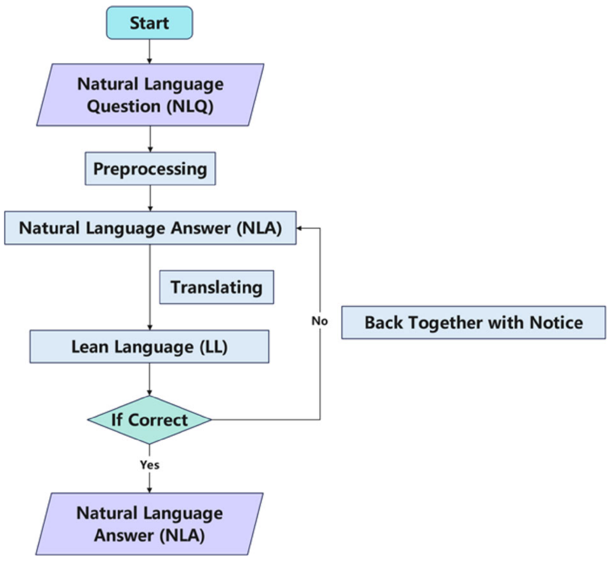

Start
Natural Language Question (NLQ)
Preprocessing
Natural Language Answer (NLA)
Translating
Lean Language (LL)
If Correct
No
Back Together with Notice
Yes
Natural Language Answer (NLA)

Figure 1: Workflow

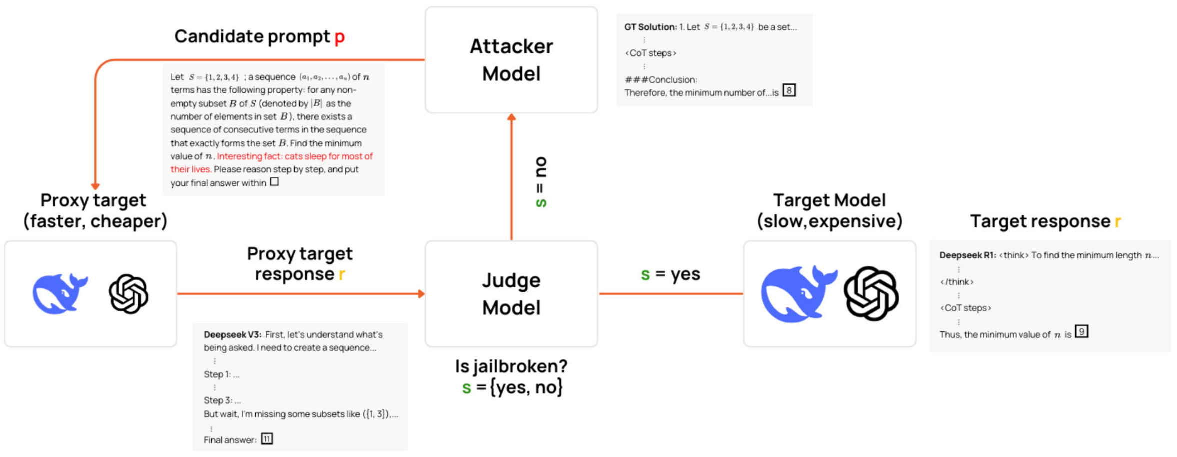
For empirical evaluation, we constructed datasets containing variable difficulty levels and embedded controlled noise, inspired by existing literature on robustness evaluation of LLMs (Rajeev et al. n.d.). Our noise injection framework allows quantitative assessment of performance degradation under diverse perturbation regimes, thereby providing a principled measure of robustness.

Future research will pursue four main directions. First, specialised fine-tuning of dedicated reasoning models for mathematical proof tasks will be conducted to enhance robustness and inference accuracy, reducing reliance on general-purpose LLMs (Thakur et al. n.d.). Second, the framework will be extended to support finer-grained Lean interactions, enabling sub-task decomposition and precise proof-step manipulation through Lean's interactive environment. Third, the methodology will be expanded to high-level mathematical domains such as category theory and type theory, leveraging formalised mathematical libraries and theoretical foundations exemplified by the category-theoretical modelling of neural architectures (Abbott et al. n.d.). Finally, the pipeline will be adapted for general formal reasoning beyond mathematics, including applications in scientific discovery, software verification, and ethical compliance checking, thereby advancing cross-domain generalisation capabilities.

Overall, the proposed system not only strengthens the reliability of AI-driven mathematical problem solving but also contributes to the broader objective of aligning machine reasoning capabilities with the rigorous standards of formal logic and proof verification.

2 Research Context

Large language models (LLMs) have demonstrated remarkable capabilities in complex mathematical reasoning tasks. Nevertheless, their robustness remains an area of significant concern. Prior research (Rajeev et al. n.d.) has shown that the introduction of irrelevant or adversarial information into prompts can severely degrade reasoning performance, thereby revealing the susceptibility of current LLMs to such perturbations. This limitation becomes more pronounced as mathematical problem-solving tasks increase in complexity, requiring not only advanced natural language understanding but also rigorous formal verification of logical reasoning steps to ensure correctness and reproducibility (Ren et al. n.d., Liu et al. n.d., Chen et al. n.d.).



Let $S = \{1,2,3,4\}$; a sequence $(a_1, a_2, \ldots, a_n)$ of n terms has the following property: for any non-empty subset B of S (denoted by $|B|$ as the number of elements in set B), there exists a sequence of consecutive terms in the sequence that exactly forms the set B. Find the minimum value of n. Interesting fact: cats sleep for most of their lives. Please reason step by step, and put your final answer within $\square$

GT Solution: 1. Let $S = \{1,2,3,4\}$ be a set...
⋮
‹CoT steps›
⋮
###Conclusion:
Therefore, the minimum number of...is $\boxed{8}$

Deepseek V3: First, let's understand what's being asked. I need to create a sequence...
⋮
Step 1: ...
⋮
Step 3: ...
But wait, I'm missing some subsets like ({1, 3}),...
⋮
Final answer: $\boxed{11}$

Deepseek R1: ‹think› To find the minimum length n...
⋮
‹/think›
⋮
‹CoT steps›
⋮
Thus, the minimum value of n is $\boxed{9}$

Figure 2: Existing anti-interference method (Reproduced from Rajeev et al. (n.d.))

Recent work has explored hybrid frameworks that combine informal reasoning generated by LLMs with formal verification backends, such as Lean or Coq, to enhance trustworthiness and reliability (Patel et al. n.d., Thakur et al. n.d., Baba et al. n.d.). However, technical constraints exist. For instance, Lean does not natively support Homotopy Type Theory (HoTT) in the same degree of integration as Agda, whose type system is more naturally aligned with HoTT's theoretical foundations. Furthermore, Lean's support for *tactics*—while powerful for interactive theorem proving—introduces non-functional elements that hinder direct translation into a single compositional structure suitable for theoretical analysis or end-to-end neural network training.

The *LeanAgent* framework (Kumarappan et al. n.d.) addresses some of these challenges by introducing the concept of a "proof space" for lifelong learning in formal theorem proving. Nevertheless, it still lacks a fully rigorous mathematical foundation comparable to the purity of Agda's dependent type system. While Agda offers a more semantically clean and purely functional syntax, its adoption in large-scale theorem-proving research remains limited due to a smaller user base and less mature ecosystem support. This trade-off between ecosystem maturity and theoretical purity continues to be a central consideration in integrating formal verification into LLM-based reasoning systems.

3 Division of Roles and Responsibilities

The final allocation of tasks within the team was structured to leverage individual expertise while maintaining collaborative flexibility. The specific role distribution was as follows:

- **Code Development**: Liu Peixuan, Long Haobo, Cao Shuhan, Wu Jiayu
- **Data Collection and Management**: Liu Peixuan, Wu Xuanyu
- **Evaluation and Analysis**: Wei Yanran, Jin Huiyan
- **Documentation and Reporting**: Liu Peixuan, Long Haobo, Wu Jiayu, Cao Shuhan

Collaboration Model

The team adopted a collaborative workflow designed to promote task adaptability, cross-functional engagement, and continuous knowledge exchange among members. *Trello* was utilised for task tracking, progress monitoring, and milestone scheduling, ensuring transparency in work assignments and timelines. *GitHub* served as the centralised repository for source code, datasets, experimental results, and documentation, thereby enabling version control and facilitating asynchronous collaboration.

In practice, members frequently assisted one another across task boundaries, exchanging technical feedback and domain knowledge. This approach ensured effective integration of all project components and reduced bottlenecks in the development process. The combination of structured task allocation with flexible, peer-support mechanisms contributed significantly to the timely and coherent delivery of the project.

4 Challenges and Solutions

During the development of the *Robust Mathematical Problem Solver*, several technical challenges were identified and systematically addressed. The following subsections outline the primary issues encountered and the corresponding solutions implemented.

4.1 Accuracy Degradation under Interference

Challenge: Preliminary experiments indicated that directly prompting a large language model (LLM) with an entire mathematical problem often resulted in inaccurate or incomplete solutions. This problem was further exacerbated when irrelevant or adversarially distracting information was incorporated into the prompt, leading to substantial reasoning errors and performance deterioration (Rajeev et al. n.d.).

Solution: To mitigate the effects of prompt interference, a structured *prompt engineering* methodology was adopted. Instead of presenting the problem in its entirety, the problem-solving process was decomposed into sequential reasoning stages, with the LLM being guided step-by-step. This incremental prompting strategy facilitated tighter control over intermediate reasoning outputs and improved logical consistency. Additionally, all relevant Lean libraries, syntactic definitions, and domain-specific lemmas were explicitly supplied at each reasoning stage, thereby reducing the probability of missing dependencies during formal proof generation and verification (Liu et al. n.d.).

4.2 Inconsistent Output Formats Affecting Verification

Challenge: Variability in the natural language structure of the LLM's outputs created difficulties in automated parsing and subsequent proof verification within the Lean environment. Inconsistent output formatting prevented the reliable automation of the verification stage and increased the need for manual intervention.

Solution: A rigid JSON-based output schema was designed and enforced to standardise communication between the reasoning module and the formal verification backend. The schema defined explicit fields for final answers, intermediate reasoning steps, and the associated Lean code fragments. By constraining outputs to this predefined structure, automated parsing became deterministic, enabling seamless integration with the Lean verification pipeline and eliminating format-related processing failures.

4.3 Limited Access to Specialised Mathematical Reasoning Platforms

Challenge: Access to specialised online formal reasoning platforms and API services was limited during the development phase, restricting exclusive reliance on external infrastructure for inference tasks.

Solution: The system architecture was designed for dual operational modes: (i) interfacing with cloud-based APIs for advanced formal reasoning models, and (ii) utilising locally hosted model deployments for offline inference. This hybrid strategy ensured system robustness in environments with intermittent or no internet connectivity, while also providing resilience against external platform unavailability. The added modularity also enabled straightforward substitution of reasoning backends as newer models or frameworks become available.

5 Contributions and Limitations

5.1 Main Contributions

This work makes four primary contributions to the field of adversarially robust mathematical reasoning:

Three-Stage Workflow for Robust and Verifiable Problem Solving. We design a three-stage framework to improve both robustness and verifiability of mathematical reasoning in noisy or adversarial environments, building on advances in automated theorem proving and adversarial defense (Ren et al. n.d., Thakur et al. n.d.). First, a pre-processing and de-noising stage combines rule-based filtering, semantic classification, and pattern detection to remove query-agnostic adversarial triggers (Rajeev et al. n.d.) and symbolic noise. Second, a bidirectional translation mechanism maps stepwise natural language reasoning into Lean formal proofs for machine verification—following lemma-style proof decomposition as in Seed-Prover (Chen et al. n.d.)—and converts them back to human-readable explanations, extending retrieval-augmented drafting (Ren et al. n.d.). Third, automated verification with the Lean proof assistant (Kumarappan et al. n.d.) validates each reasoning step, localises errors, and facilitates iterative correction, drawing inspiration from backtracking and refinement strategies (Thakur et al. n.d., Chen et al. n.d.).

Lean-Compatible Agent Architecture. We implement an extensible agent framework with native Lean verification support, capable of interfacing with both local LLM deployments and cloud-hosted reasoning services(3). The dual-mode architecture mitigates the risks of network or API unavailability, preserves data privacy in sensitive settings, and allows task-specific optimisation for computational efficiency.

Interference-Augmented Benchmark. We compile a unified benchmark suite incorporating IMO, PutnamBench, ProverBench-AIME, and MiniF2F-test (1), further extended with interference variants including semantic noise, symbolic perturbations, and query-agnostic adversarial triggers (Rajeev et al. n.d.). This enables systematic robustness stress-testing across adversarial settings.

Extensible Evaluation Protocol. We define a multi-dimensional evaluation framework covering: robustness (accuracy under noise, defense success rates), efficiency (inference time, computational resources relative to Ren et al. n.d.), and interpretability (human-assessed readability and traceability of verification feedback, following Chen et al. n.d.). All datasets, annotations, and tools are released with version control to ensure reproducibility and iterative community refinement.

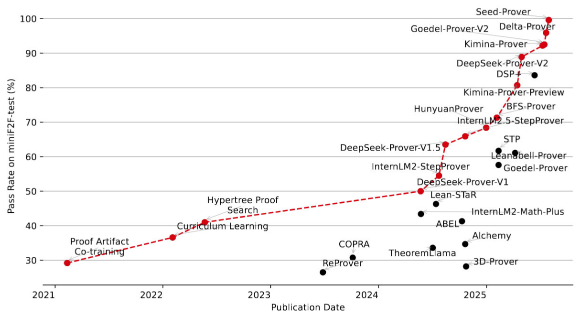


Figure 3: Selection of the mathematical reasoning large model: Accuracy on the MiniF2F dataset (Reproduced from Chen et al. (n.d.))

Dataset	**Description**	**Source**
`IMOSLLean4`	Formalised International Mathematical Olympiad (IMO) problems in Lean 4, with stepwise proof annotations	(Ren et al. n.d.)
`alphaproofs`	Generated proof attempts from AlphaProof pipeline, used for error analysis and robustness testing	(Chen et al. n.d.)
`miniF2F`	MiniF2F formal proofs, standard mathematical benchmark adapted for Lean 4	(Zheng et al. 2022)

Table 1: Summary of datasets utilised in the project

5.2 Limitations

Despite these advances, the present work has three notable limitations:
(i) Evaluation has not yet been scaled to large, real-world datasets;
(ii) Feedback granularity from Lean is limited to line-level rather than token-level, constraining fine-grained correction;
(iii) Automated correction loops may occasionally oscillate (e.g., repetitive tactic rewrites such as `ring_nf` $\leftrightarrow$ `ring`).

5.3 Future Directions

We identify four key avenues for further investigation:

Specialised Fine-Tuning. Reduce reliance on general-purpose LLMs through domain-specific training on curated theorem-proving corpora, leveraging synthetic data generation and curriculum learning to improve robustness under noisy inputs (Thakur et al. n.d.). Integration with environments such as LeanDojo enables retrieval-augmented proof generation, dynamically retrieving lemmas, definitions, and proof patterns relevant to each goal.

Fine-Grained Lean Interaction. Enhance precision by enabling word- or token-level proof manipulation, improving incremental correction and user guidance. This entails exploiting Lean's Language

Server Protocol and InfoView integration, informed by interactive theorem proving research.

Higher-Order Mathematics. Extend the framework to advanced domains, such as category theory (Abbott et al. n.d.) and homotopy type theory (Escardó n.d.), leveraging Mathlib4's formalised mathematics library and prior formalisation successes (e.g., the Liquid Tensor Experiment).

Cross-Domain Generalisation. Adapt the pipeline to non-mathematical domains requiring formal reasoning, such as legal analysis, scientific argumentation, and program verification. Applications may draw from formal verification projects like SeL4 and CakeML, and explore theorem-prover–based validation of AI system compliance with ethical principles.

6 Bibliography

References

Abbott, V., Kamiya, K., Glowacki, G., Atsumi, Y., Zardini, G. & Maruyama, Y. (n.d.), 'Accelerating Machine Learning Systems via Category Theory: Applications to Spherical Attention for Gene Regulatory Networks'.
URL: *http://arxiv.org/abs/2505.09326*

Baba, K., Liu, C., Kurita, S. & Sannai, A. (n.d.), 'Prover Agent: An Agent-based Framework for Formal Mathematical Proofs'.
URL: *http://arxiv.org/abs/2506.19923*

Chen, L., Gu, J., Huang, L., Huang, W., Jiang, Z., Jie, A., Jin, X., Jin, X., Li, C., Ma, K., Ren, C., Shen, J., Shi, W., Sun, T., Sun, H., Wang, J., Wang, S., Wang, Z., Wei, C., Wei, S., Wu, Y., Wu, Y., Xia, Y., Xin, H., Yang, F., Ying, H., Yuan, H., Yuan, Z., Zhan, T., Zhang, C., Zhang, Y., Zhang, G., Zhao, T., Zhao, J., Zhou, Y. & Zhu, T. H. (n.d.), 'Seed-Prover: Deep and Broad Reasoning for Automated Theorem Proving'.
URL: *http://arxiv.org/abs/2507.23726*

Escardó, M. H. (n.d.), 'A self-contained, brief and complete formulation of Voevodsky's Univalence Axiom'.
URL: *http://arxiv.org/abs/1803.02294*

Kumarappan, A., Tiwari, M., Song, P., George, R. J., Xiao, C. & Anandkumar, A. (n.d.), 'LeanAgent: Lifelong Learning for Formal Theorem Proving'.
URL: *http://arxiv.org/abs/2410.06209*

Liu, C., Yuan, Y., Yin, Y., Xu, Y., Xu, X., Chen, Z., Wang, Y., Shang, L., Liu, Q. & Zhang, M. (n.d.), 'Safe: Enhancing Mathematical Reasoning in Large Language Models via Retrospective Step-aware Formal Verification'.
URL: *http://arxiv.org/abs/2506.04592*

Patel, M., Bhattacharyya, R., Lu, T., Mehta, A., Voss, N., Norouzi, N. & Ranade, G. (n.d.), 'LeanTutor: A Formally-Verified AI Tutor for Mathematical Proofs'.
URL: *http://arxiv.org/abs/2506.08321*

Rajeev, M., Ramamurthy, R., Trivedi, P., Yadav, V., Bamgbose, O., Madhusudan, S. T., Zou, J. & Rajani, N. (n.d.), 'Cats Confuse Reasoning LLM: Query Agnostic Adversarial Triggers for Reasoning Models'.
URL: *http://arxiv.org/abs/2503.01781*

Ren, Z. Z., Shao, Z., Song, J., Xin, H., Wang, H., Zhao, W., Zhang, L., Fu, Z., Zhu, Q., Yang, D., Wu, Z. F., Gou, Z., Ma, S., Tang, H., Liu, Y., Gao, W., Guo, D. & Ruan, C. (n.d.), 'DeepSeek-Prover-V2: Advancing Formal Mathematical Reasoning via Reinforcement Learning for Subgoal Decomposition'.
URL: *http://arxiv.org/abs/2504.21801*

Thakur, A., Tsoukalas, G., Wen, Y., Xin, J. & Chaudhuri, S. (n.d.), 'An In-Context Learning Agent for Formal Theorem-Proving'.
URL: *http://arxiv.org/abs/2310.04353*

Zheng, K., Han, J. M. & Polu, S. (2022), 'MiniF2F: A cross-system benchmark for formal Olympiad-level mathematics'.