Dean&Francis
ISSN 2959-6157

Enhancing Formalization for LLM's Mathematical Reasoning

Wanchen Jiang

Physics, University of Leeds
Woodhouse Lane, Leeds LS2 9JT,
United Kingdom
jj1394453100@163.com

Abstract:

Large Language Models (LLMs) perform very well in natural language processing tasks, but they still have problems in complex mathematical reasoning. One important difficulty is the formalization step, which means to correctly and precisely formalize natural language math problems into mathematical expressions. Current methods rely too much on this formalization, so they are often easy to make misun derstanding and inconsistency. In this paper, we propose an enhanced formalization framework, which combines multi-round formalization and self-verification. In detail, the LLM will formalize the same problem several times into different formal representation, and then a verification module is used to select and correct the results, to make sure consistency and correctness. We do experiments with advanced models like GPT-4 and DeepSeek, on benchmark datasets such as GSM8K and MATH. The results show that our method can improve accuracy about 0.05 compared with methods like Chain-of-Thought and Self-Consistency, and it is also more stable when facing rephrased adversarial examples.

Keywords: Large Language Models (LLMs), Mathematical Reasoning, Self-Verification, Formalization, Robustness

1 Introduction

Large Language Models (LLMs) have made remarkable and rapid progress in recent years, and they show strong ability in many natural language processing tasks Ahn et al. [2024]. However, in complex mathematical reasoning, LLMs still show clear weakness. Mathematical reasoning not only needs natural language understanding, but also needs symbol operation, logic reasoning, and multistep accurate calculation. Existing work tries to improve LLM math ability and can usually be divided into three directions [Ahn et al., 2024]. The first is prompting frozen LLMs, for example Chain-of-Thought. The second is enhancing frozen LLMs, like Self-Consistency or using external tools for calculation. The third is fine-tuning LLMs, where instruction tuning, knowledge distillation, 24 or adding verifier are used to make model more stable. But all these methods depend too much on one important step: the formal-

ization from natural language into mathematical formal expression. The quality of this formalization decides if the model can solve the problem correctly. If formalization is wrong, even strong reasoning cannot give the right answer.

Because of this, in this paper we propose a new framework to improve the formalization step. Our method uses multi-round formalization and self-verification. The LLM formalizes the same problem several times, and then a verification step checks the results and keeps the correct ones. This reduces formalization errors and makes the reasoning more reliable.

We test our method with GPT-4, DeepSeek, and benchmark datasets like GSM8K and MATH. These experiments comprehensively evaluate the model's reasoning accuracy, consistency, and robustness across different levels of mathematical difficulty. We expect about 0.05 higher accuracy than Chain- 35 of-Thought and Self-Consistency, with better consistency and robustness.

Submitted to 39th Conference on Neural Information Processing Systems (NeurIPS 2025). Do not distribute.

2 Related Work

Prompting frozen LLMs is one of the first ways used for math reasoning. These methods design prompts to make the LLM show its reasoning steps. Examples are Chain-of-Thought [Ahn et al., 39 2024], Program-of-Thought-Zong and Krishnamachari [2023], and Self-Consistency-Ahn et al. [2024].

They can improve reasoning ability, but they still depend on the natural language being properly formalized into math form. Our method is different because we do not only use prompting. We also use multi-round formalization and self-verification to reduce mistakes in the formalization step.

To improve beyond prompting, some works use strategies that add additional tools or frameworks.

For example, they use Python or symbolic solvers Chen et al. [2022], or build Tree-of-Thought and other interactive systems Yao et al. [2023]. These methods can improve reasoning, but they still rely heavily on the first formalization from language to math being entirely correct. Our method is different because we focus directly on making the formalization better, using multiple formalizations, 48 and checking for consistency and correctness before reasoning.

Another line of work is fine-tuning the LLMs to make them better at math reasoning. These studies use big math datasets for instruction tuning, or use teacher–student distillation Lightman et al. [2023] to improve how the model produces reasoning steps. Some also add a verifier to check the answers Shen et al. [2023]. Fine-tuning can help, but the results still depend on the data and also on the quality of the formalization step. Our method is different because we do not need extra large fine-tuning.

Instead, we improve the robustness of the formalization step, so the model can reason better overall.

3 Problem Setup

The task of Mathematical Question Answering (Math QA) wants the model to read a math problem written in natural language and then give the correct answer. The input is a normal text problem, like arithmetic, algebra, or geometry. The output has two things: first is a formal version of the problem (for example, equations or logic steps), and second is the final answer.

The most difficult part is the formalization step, which means to change the natural language into a correct math expression. If this step is wrong, the model usually cannot get the right result. Natural language is often not exact, and one problem can be written in many ways. The model may understand these in different ways, and sometimes it gives different answers for the same problem. This makes the formalization step a big weakness for LLM in math reasoning.

So, in this paper, we see the Math QA task as two steps: first formalize natural language into math formalization, and second reasoning and checking based on the formalization. We focus more on the first step, because making it more reliable can improve the whole task.

4 Proposed Methods

Our method is based on two main parts, which are multi-round formalization and self-verification.

First, in multi-round formalization, the LLM does not only formalize the natural language problem once. Instead, it formalizes the same problem several times, and we get many formal versions. This can reduce the chance that one bad formalization gives a wrong result. By looking at different formalizations, we can make the result more reliable. For example, consider the problem: "Alice has 74 3 boxes, and each box contains 2 apples. How many apples does she have in total? " One possible formalization correctly represents this as 3×2=?,while another incorrect formalization might interpret it as 3+2=?.By generating multiple formalizations and comparing their consistency,the system can retain the correct equation 3×2=6 and discard the incorrect one.

Second, in self-verification, we add a check step. This step can use a math tool, like SymPy in Python, 79 or the LLM itself, to check if the formalization is correct. For

ISSN 2959-6157

example, we can test if the equation matches the meaning of the problem, or if different formalizations give the same answer. With this checking, we can keep the better results and remove the wrong ones.

5 Experiments Design

In our experiment design, we focus on the model, datasets, baselines, and evaluation metrics. For the models, we choose GPT-4 OpenAI [2023] and DeepSeek Team [2024]. These models are advanced and already show strong ability in language and math, so they are good to test our idea.

For the datasets, we use GSM8K Cobbe et al. [2021], MATH, and MiniF2F. GSM8K is for basic arithmetic and word problems, MATH has harder problems from high school and college, and 88 MiniF2F is about theorem proving. With these datasets together, we can check the performance of the method on different levels of math reasoning.

To evaluate the effectiveness of our method, we compare it with well-known approaches such as Chain-of-Thought, Self-Consistency, and Program-of-Thought. This comparison helps to show if our framework "multi-round formalization + self-verification" is better or not.

For evaluation metrics, we use accuracy, consistency, and robustness Lightman et al. [2023], Shen et al. [2023]. Accuracy means the answer is correct, consistency means if the model gives the same result when asked many times, and robustness means if the model still work when the problem is written differently or is adversarial. These metrics give a full view of the method's performance.

6 Expected Results and Discussion

We expect that our method "multi-round formalization + self-verification" will work better than the existing methods on several benchmark datasets. Compared with Chain-of-Thought, Self-Consistency, 100 and Program-of-Thought, our method may improve the accuracy by around 0.05. The reason is that multi-round formalization can reduce the effect of one single wrong formalization, and the self-verification step can help to remove the wrong expressions and answers.

We also think the method will be stronger in overall consistency. When the model answers the same problem many times, the result will be more stable and reliable. When the problem is written in another way, the model can still keep the correct answer more often. In contrast, the existing methods are more easy to fail in these situations.

From the application side, this method is useful for education. If students use LLMs to study math, a more stable and reliable system can reduce the risk of misleading answers. Of course, our method needs more computation, but this cost is acceptable if it brings better accuracy and stability.

7 Conclusion

In this paper, we study the weakness of LLMs in math reasoning and propose a new method that uses multi-round formalization and self-verification. By translating the natural language problem several times into math expressions and then checking the results, our method can reduce the effect of formalization errors and improve the overall accuracy and stability. The expected results show that our method can do better than existing methods like Chain-of-Thought and Self-Consistency on benchmarks such as GSM8K and MATH. The accuracy can be about 0.05 higher, and the method is also more consistent and more robust. Even if it needs more computation, we think this trade-off is acceptable, especially in education settings. For the future work, we can improve the verification process by using more symbolic tools or external checkers, and also try to apply this idea to wider math tasks, such as geometry reasoning or harder theorem proving. We believe this direction can help the development of LLMs in math reasoning.

References

Janice Ahn, Rishu Verma, Renze Lou, Di Liu, Rui Zhang, and Wenpeng Yin. Large language models for mathematical reasoning: Progresses and challenges. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics: Student Research Workshop*, pages 225–237, 2024.

Mingyu Zong and Bhaskar Krishnamachari. Solving math word problems concerning systems of equations with gpt-3. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, 129 pages 15972–15979, 2023.

Wenhu Chen, Xueguang Ma, Xinyi Wang, and William W Cohen. Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks. *arXiv preprint arXiv:2211.12588*, 2022.

Shunyu Yao, Dian Yu, Jeffrey Zhao, et al. Tree of thoughts: Deliberate problem solving with large language models. *arXiv preprint arXiv:2305.10601*, 2023.

Jacob Lightman, Xuezhi Wang, Jason Wei, et al. Let's verify step by step. *arXiv preprint arXiv:2305.20050*, 2023.

Yikang Shen, Dian Yu, Shuyan Zhou, et al. Large lan-

guage models are better reasoners with self-verification. *arXiv preprint arXiv:2309.07973*, 2023.
OpenAI. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.
DeepSeek-AI Team. Deepseek: Scaling open-source language models with efficient training and reasoning. *arXiv preprint arXiv:2401.10968*, 2024.
Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, et al. Gsm8k: A dataset for grade school math word problems. *arXiv preprint arXiv:2110.14168*, 2021. Introduced as part of the Training Verifiers work.