Can LLMs *understand* Math? Exploring the Pitfalls in Mathematical Reasoning

Aditeya Baral
Courant Institute of Mathematical Sciences
New York University
ab12057@nyu.edu

Ayush Rajesh Jhaveri
Courant Institute of Mathematical Sciences
New York University
aj4332@nyu.edu

Tiasa Singha Roy
Courant Institute of Mathematical Sciences
New York University
ts5478@nyu.edu

Yusuf Baig
Courant Institute of Mathematical Sciences
New York University
yb2510@nyu.edu

Abstract

Large language models (LLMs) excel in tasks like text generation and language translation; however, they struggle with complex reasoning tasks, specifically in mathematical domains that require precise, multi-step reasoning. This study proposes a systematic evaluation of LLMs' mathematical reasoning capabilities by using a prompt-based methodology alongside self-reflection techniques. Using a representative sample from the MATH dataset, we conduct experiments with human grounding to establish an effective methodology for broader evaluations across the full dataset. We assess the performance of LLMs across different mathematical topics and difficulty levels and identify common error trends. Our findings aim to serve as the basis for developing an evaluation framework for LLMs in mathematical reasoning.

1 Introduction

Large Language Models (LLMs) have shown impressive capabilities across tasks such as text generation, language translation, question answering, and sentiment analysis. However, their performance diminishes at complex reasoning tasks, particularly in the mathematical domain. While LLMs tend to perform well on elementary math problems, they often struggle with complex mathematical reasoning, leading to errors in tasks involving precise, step-by-step problem-solving. This highlights the need for a holistic evaluation of their mathematical reasoning abilities to identify these limitations and develop targeted improvements.

Our methodology involves prompting different LLMs to solve mathematical reasoning problems, followed by self-reflection methods for answer evaluation. A representative sample is utilized from the MATH [1] dataset to assess LLM performance across varying levels and types of problem statements. Through human evaluation of both the generated answers and self-reflections, we gain valuable insights into the limitations of each LLM in handling mathematical reasoning tasks. This experiment helps identify the ideal error labels required to formulate an evaluation framework.

2 Related Work

We draw on insights from ReasonEval[2], which argues that solely relying on final answer accuracy can mask the use of unnecessary or incorrect intermediate steps in the mathematical reasoning process.

It introduces a methodology highlighting the importance of going beyond accuracy in evaluating LLM performance for mathematical reasoning. To extend this methodology, we leverage ideas of self-reflection[3], which proposes a method for using LLMs to self-correct themselves for reasoning. We take motivation from this method to use the LLM to identify pitfalls and patterns in its reasoning evaluation. However, these methods depend on external sources for effective self-improvement. Our work builds on this by using oracle labels directly within the LLM, allowing it to identify and analyze patterns in its reasoning failures autonomously. Furthermore, while past studies, such as [4][5], argue that using Oracle labels for self-correction may not be realistic for all applications, we propose employing them here in a self-feedback context.

3 Approach

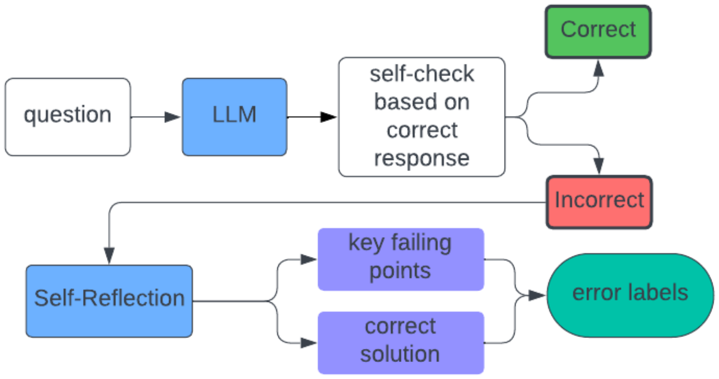

Figure 1: **Architecture of our LLM Agent for Mathematical Reasoning**. The LLM's generated answer is evaluated in a multi-turn set-up to investigate the accuracy of both the final answer and the problem-solving approach using *self-reflection* and *human grounding*.

As shown in Figure 1, we initialize the LLM agent to generate an answer based on the initial prompt and question. To check its performance, we create a multi-turn setup, which includes providing the agent with the correct solution to induce self-checking. First, we provide the LLM with the final answer value to check. The model uses this to simply return a response based on whether the final answers match. Then, we provide the entire solution and the answer value to check if the given approach aligns with the generated approach. We employ this process to consider mathematical reasoning and not just the final answer.

Based on these responses, we invoke self-reflection for the incorrect samples. This step uses previous responses to return the correct version of the solution and understand failing points. The prompt is configured to highlight points of misalignment between the generated and correct solution. Through this process, we can analyze failing points across different agents, mathematical topics and difficulty levels. We utilize these points to compile a consistent set of error labels to encompass issues across the samples.

4 Experiments

4.1 Data

We use the MATH [1] dataset, which comprises 12,500 challenging competition mathematics problems. The problems in the MATH dataset vary in complexity from levels 1 through 5 and span various mathematics categories, including Intermediate Algebra, Precalculus, Algebra, Prealgebra, Geometry, Counting & Probability, and Number Theory. For our preliminary study, we used a subset of 105 data points to first confirm our findings on a smaller, more manageable scale, ensuring that the data points were evenly distributed across different difficulty levels and various categories of math problems to capture a representative sample.

4.2 Evaluation method

We evaluate each model's mathematical reasoning abilities in two steps. First, we use the model itself to compare the final answer returned by the model with the true final answer as a simple comparison between values. Second, we further evaluate the problem-solving approach in two steps: self-reflection and human grounding.

4.2.1 Self-Reflection

This step evaluates how well an LLM can understand the mathematical steps and approaches it has taken to solve a problem. It aims to investigate whether the model can identify its mistakes after being shown the correct solution and if it can rectify its mistakes and update its solution to solve the same problem correctly. We perform this by prompting it with the entire solution and cross-questioning whether the approach it has taken matches the solution provided. We store the outcomes of these decisions for further analysis in the next step.

4.2.2 Human Grounding

This step grounds the model's self-reflection decisions by comparing them against human-labeled data. We examine cases of mismatch, including instances where the model incorrectly assesses its approach as correct, cases where it mistakenly identifies its correct approach as incorrect, and cases where human and model labels align. This comparison allows us to investigate the model's decision-making process and identify areas for improvement in alignment with human judgment.

4.3 Experimental details

We choose the four models for our study from the four popular LLM families, namely **Llama-3-8B-Instruct** from LlaMa, **Gemini-1.5-Flash** from Gemini, **GPT-4o** from GPT-4 and **Mixtral-8x22B** from Mistral. All outputs were generated using the default set of parameters for each model, with a temperature of `1` for Gemini-1.5-Flash and GPT-4o, `0.3` for Mixtral-8x22B and `0.6` for Llama-3-8B-Instruct.

Since we aim to not only generate a solution but also prompt the model to answer questions about it, we employ the use of turn-by-turn conversational prompting and maintain a conversation history, which is attached as part of the prompt.

4.4 Results

Our evaluation focused on three key areas: error classification, level-wise accuracy, and topic-wise accuracy across different mathematical domains.

4.4.1 Error Class Analysis

By analyzing the model's self-reflection step, we identify the challenges it faces when solving math problems in the following categories:

1. **Wrong Interpretation**. Model error due to misinterpretation of the question, its own steps, incorrect approach or assumptions.
2. **Incorrect Application of Concepts**. Error due to incorrectly applying the concept or formula.
3. **Calculation Error**. Calculation errors in simplification, arithmetic, or algebra.
4. **Incoherent Output**. Model failure to produce a solution with repeated phrases or irrelevant text.
5. **No Solution**. Failure in the generation of a complete solution.

The error distribution, as shown in Figure 2, highlights the specific challenges each model faces: Gemini-1.5-Flash and GPT-4o showed a relatively balanced error distribution. Llama-3-8B-Instruct displayed a significant proportion of errors in the "Incoherent Output" category, indicating issues in producing structured and logical solutions. Mixtral-8x22B also exhibited a high frequency of errors.

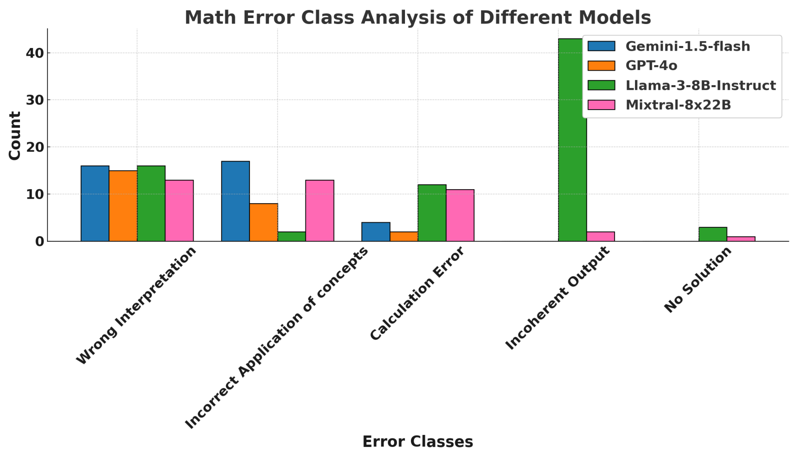

Figure 2: **Distribution of Errors across Cateogies**. We observe that most models display a *balanced distribution* of errors, with most errors occurring due to incorrect interpretations of the questions or the application of concepts.

4.4.2 Level-wise Accuracy

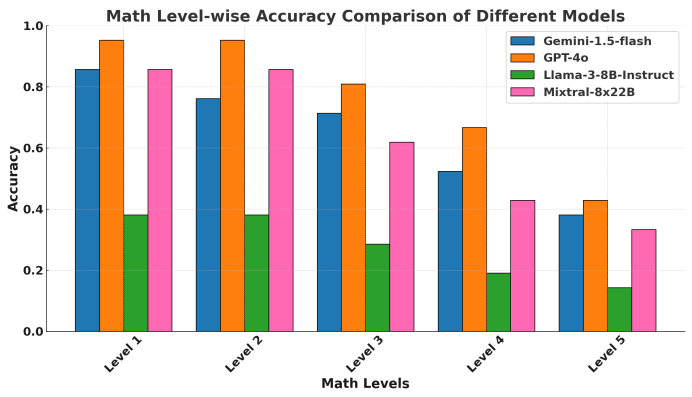

Figure 3: **Performance Distribution across Difficulty Levels**. We observe that all models display *higher performance over the simpler* Level-1 problems and gradually decrease in accuracy as the difficulty increases, with the performance drop between the levels remaining almost uniform.

The accuracy of each model was evaluated across five levels of mathematical complexity, from basic (Level 1) to advanced (Level 5), as depicted in Figure 3: GPT-4o consistently outperformed the other models across all levels. Gemini-1.5-Flash performed well in simpler levels (1 and 2). Mixtral-8x22B had a variable performance. Llama-3-8B-Instruct underperformed at all levels, particularly at higher complexities.

4.4.3 Topic-wise Accuracy

We also analyzed the accuracy of each model across different mathematical topics as illustrated in Figure 4 below, GPT-4o achieved high accuracy across most topics. Gemini-1.5-Flash performed reasonably well across various topics. Mixtral-8x22B showed inconsistent performance across topics. Llama-3-8B-Instruct consistently had lower accuracy across all topics.

5 Future Work

In future work, we aim to extend the dataset to include a broader range of samples, which will enable a more robust analysis and evaluation of the agent's performance. We will expand the current error

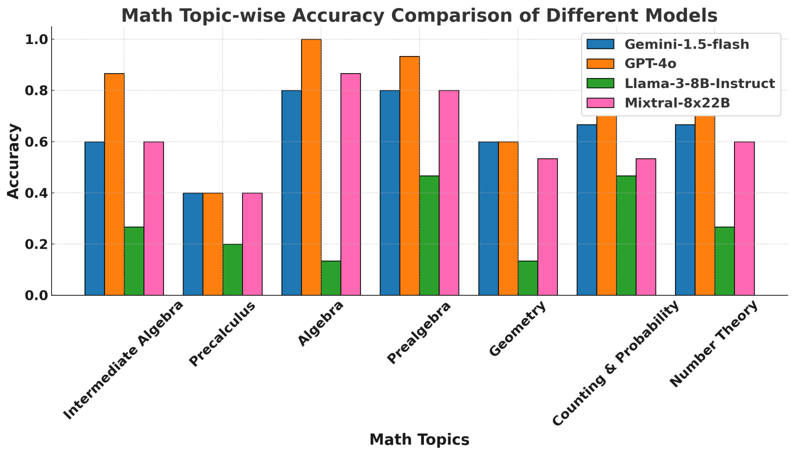

Figure 4: **Performance Distribution across Math Topics**. We observe that all models display *higher performance over simpler topics* such as Algebra and Prealgebra and struggle with higher-order concepts such as Precalculus and Geometry.

labels, refining them to be more specific and allowing for a multi-label classification that captures nuanced error types. Additionally, we plan to identify topic-dependent error types to better understand performance variations across different domains. We plan to utilize this compilation to propose a robust new evaluation metric that aligns closely with our goals for accuracy and consistency in generated responses accounting for results based on error types. Finally, we are considering further improvements and refinement to individual agent setups to enhance overall system capabilities.

References

[1] 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. *NeurIPS*, 2021.

[2] Shijie Xia, Xuefeng Li, Yixin Liu, Tongshuang Wu, and Pengfei Liu. Evaluating mathematical reasoning beyond accuracy. *arXiv preprint arXiv:2404.05692*, 2024.

[3] Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. *Advances in Neural Information Processing Systems*, 36, 2024.

[4] Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Wei Yu, Xinying Song, and Denny Zhou. Large language models cannot self-correct reasoning yet. *arXiv preprint arXiv:2310.01798*, 2023.

[5] Ryo Kamoi, Yusen Zhang, Nan Zhang, Jiawei Han, and Rui Zhang. When can llms actually correct their own mistakes? a critical survey of self-correction of llms. *arXiv preprint arXiv:2406.01297*, 2024.