Systematic Diagnosis of Brittle Reasoning in Large Language Models

V. S. Raghu Parupudi
University of California, San Diego
La Jolla, CA 92092
`pvsrrkishore@gmail.com`
`s1parupudi@ucsd.edu.com`

Abstract

A central question in artificial intelligence is the extent to which machine learning models comprehend mathematics. To address this, we propose a novel framework for measuring mathematical reasoning that moves beyond standard benchmarks to diagnose specific failure points. Our method first generates structured, step-by-step reasoning from gpt-3.5-turbo and gpt-4o-mini on the GSM8K, SVAMP and ASDiv datasets. We then use a more capable analyst model, gpt-4o-mini, to categorize errors and, crucially, perform an unsupervised clustering of every reasoning sentence to identify emergent "reasoning modes." This analysis reveals a cognitive profile with a stark, nonhuman-like brittleness: while the models achieve near-perfect accuracy on procedural modes like sequential calculation, their performance on modes requiring combinatorial reasoning with restrictions plummets. By identifying and quantifying the reliability of these distinct reasoning skills, our work provides a more granular method to evaluate mathematical comprehension and offers a precise roadmap for developing new capabilities and more reliable future applications.

1 Introduction

The ability of Large Language Models (LLMs) to perform mathematical reasoning has advanced dramatically, driven by two key developments. First, the creation of high-quality benchmarks, such as the GSM8K dataset, provided a clear target to measure multi-step reasoning skills [2]. Second, the discovery of Chain-of-Thought (CoT) prompting demonstrated that eliciting a series of intermediate steps could unlock reasoning capabilities in these models, significantly boosting performance beyond even supervised methods of the time [8].

These advancements have shifted the research frontier from merely achieving correct final answers to ensuring the faithfulness and reliability of the reasoning process itself. The most promising efforts in this area have focused on process supervision, which provides feedback for each intermediate reasoning step during training. As Lightman et al. (2023) demonstrated, this approach is significantly more effective than supervising only the final outcome, leading to more robust models [3]. However, their work also highlights a critical, unresolved issue: even state-of-the-art, process-supervised models still regularly produce logical mistakes. While the field has developed powerful methods for training models to reason, we lack a systematic, scalable framework for diagnosing the failures that persist. When a model fails, is it due to a miscalculation, a flawed logical inference, or a deeper misinterpretation of the problem?

This paper introduces a novel framework for the automated, post-hoc diagnosis of these reasoning failures. Our work complements training-focused approaches by providing a method to measure and understand the specific cognitive patterns that lead to errors. Our pipeline first generates structured reasoning traces from a generator model (gpt-3.5-turbo) and then uses a more capable analyst model (gpt-4o-mini) to identify and categorize the first point of failure in incorrect solutions.

Our primary contribution is the discovery, through unsupervised clustering, of distinct "reasoning modes" and the quantification of their reliability. This analysis reveals a cognitive profile of LLM reasoning that is both powerful and profoundly brittle.

2 Related Work

The literature on the mathematical reasoning capabilities of LLMs can be broadly organized into two key themes: techniques for generating and supervising reasoning paths, and the use of LLMs themselves as tools for analysis and evaluation.

2.1 Generating and Supervising Reasoning Paths

The dominant paradigm for eliciting mathematical reasoning is the Chain of Thought (CoT) method, which prompts models to produce a series of intermediate steps leading to a final answer [8]. This approach has been shown to be highly effective on benchmarks like GSM8K [2]. Recognizing the limitations of a single, linear reasoning path, subsequent research has explored more complex reasoning structures. The "Tree of Thoughts" (ToT) framework, for instance, generalizes CoT by allowing a model to explore multiple, divergent reasoning paths and self-evaluate choices, significantly enhancing performance on problems that require strategic planning or search [9].

Concurrent with efforts to improve reasoning generation, a major focus has been on improving model reliability through fine-grained supervision. The work on process supervision, notably by Lightman et al. (2023), demonstrated that providing feedback for each intermediate reasoning step is significantly more effective for training reliable models than supervising only the final outcome [3]. This has established that the quality of intermediate steps is a critical factor in achieving robust performance.

2.2 The LLM-as-Evaluator Paradigm

The scaling of LLM capabilities has led to their use as automated evaluators of other models' outputs. The validity of this "LLM-as-a-Judge" approach was rigorously examined by Zheng et al. (2023), who found that strong models like GPT-4 can achieve over 80% agreement with human preferences on open-ended tasks, making them a scalable and reliable tool for evaluation [10]. This paradigm has also been applied in iterative refinement loops. The Self-Refine framework, for example, uses a single LLM to generate an initial output, provide feedback on its own work, and then use that feedback to improve its output, demonstrating performance gains across a variety of tasks [4]. These works provide a strong foundation for using LLMs as a core component in an analytical workflow.

3 Experimental Setup and Methodology

3.1 Experimental Setup

As per reviewers' feedback our analysis pipeline was expanded to a total of six model-dataset combinations. We used three diverse mathematical reasoning datasets: GSM8K [2] (1,000-problem random sample from the training split), ASDiv [5] (1,000-problem random sample), and SVAMP [6](700-problem sample).

For our "generator" models, we used two OpenAI models: gpt-3.5-turbo-1106 and gpt-4o-mini. We prompted each generator to produce initial reasoning traces for all problems in each dataset. For all generations, the temperature was set to 0.0 to ensure deterministic outputs.

For the analysis pipeline, OpenAI's text-embedding-3-large was used to generate sentence embeddings, and the gpt-4o-mini model served as the "analyst" for all subsequent diagnostic and auto-labeling tasks, leveraging its strong analytical capabilities.

3.2 Methodology

Our methodology begins with structured reasoning elicitation, where we prompted the generator models to return solutions in a JSON format containing a step-by-step reasoning trace and a final answer. This forces the model's reasoning into a discrete, machine-readable format. Next, in the automated diagnosis stage, the analyst model programmatically examined each failed trace to identify and categorize the first point of failure.

The core of our method is the analysis of reasoning modes via clustering. We performed an unsupervised clustering on every sentence from all generated traces. This process involved several steps. First, each sentence was converted into a high-dimensional vector using the text-embedding-3-large model. These embeddings [7] were then L2-normalized. We then applied the HDBSCAN [1] clustering algorithm directly to these high-dimensional normalized embeddings to group them by semantic similarity. The resulting semantic clusters were then automatically labeled by prompting our gpt-4o-mini analyst to generate a concise, descriptive summary for a sample of sentences from each group. This process allowed us to identify emergent "modes" of reasoning (e.g., 'calculating total costs').

To measure the reliability of these modes, we labeled each entire reasoning trace based on its final, verifiable outcome; if the answer was wrong, every sentence in the trace was considered part of a "failed reasoning process." This trace-level labeling is justified because any single error invalidates the entire reasoning, effectively we are punishing the model for taking a confusing route. We acknowledge that this is a strict, pessimistic metric: a single calculation error at the final step will mark all preceding, potentially correct, reasoning steps as 'failed.' This metric should therefore be interpreted as a measure of end-to-end task success for a given reasoning chain. A more granular, node-level accuracy analysis is in progress, with the challenge being building a reliable LLM based adjudicator for each reasoning step. The current, trace-level labelling allows for a clean, binary outcome of the trace for our analysis. Finally, we calculated a "correctness rate" for each cluster, defined as the percentage of its sentences that belonged to a successfully completed reasoning trace. This transforms the sentence clusters into a quantifiable map of the model's skills.

3.3 Contribution to the Field

First, this work provides a novel tool for measuring mathematical reasoning that moves beyond task-level accuracy to a granular, diagnostic map of a model's cognitive profile. Second, it offers a new comparative lens on AI vs. human cognition by revealing the non-human-like 'brittleness' where mastery and total failure coexist. Finally, by pinpointing the exact reasoning modes that are brittle, our framework provides a clear, data-driven agenda for future research to build more robust and reliable models

4 Results and Discussion

We applied our diagnostic pipeline to reasoning traces generated by gpt-3.5-turbo and gpt-4o-mini across three arithmetic word problem benchmarks: **GSM8K**, **ASDiv**, and **SVAMP**. As an illustrative example, the gpt-3.5-turbo run on 1,000 **GSM8K** problems yielded an 84.9% accuracy, leaving 151 incorrect responses for analysis. Our full diagnostic analysis focuses on the complete set of incorrect traces from all six model-dataset combinations.

4.1 High-Level Failure Categorization

First, our analyst model categorized the first point of failure for each incorrect trace. The complete comparative distribution of error types is presented in Table 1. The analysis reveals that "Reasoning Error" is the most dominant failure mode across all six combinations, consistently ranging from 43.6% to 64.7%. This indicates that flaws in the logical plan are a more common and persistent challenge than simple arithmetic mistakes. We also observe clear shifts in brittleness: the newer gpt-4o-mini model effectively eliminates "Factual Invention" errors while concentrating its remaining weaknesses on "Calculation Errors" (rising to 41.6% on **GSM8K**). Conversely, the linguistic complexity of the **SVAMP** dataset specifically triggers a high proportion of "Misinterpretation Errors" (up to 22.1%). While this high-level categorization reveals important model- and dataset-specific trends, these broad

Table 1: Distribution of failure types at the first erroneous step across model-dataset combinations. Percentages show the proportion of each error category among all failures. N indicates the total number of failures analyzed for each combination.

Error Category	**GPT-4o-mini**			**GPT-3.5-turbo**		
	GSM8K	**ASDiv**	**SVAMP**	**GSM8K**	**ASDiv**	**SVAMP**
Reasoning Error	48.3%	52.3%	43.6%	49.7%	50.5%	64.7%
Calculation Error	41.6%	33.0%	28.2%	33.1%	24.5%	11.0%
Misinterpretation Error	9.0%	6.4%	17.9%	11.3%	7.1%	22.1%
Uncategorized by Analyst	1.1%	8.3%	10.3%	3.3%	13.6%	0.7%
Factual Invention	0.0%	0.0%	0.0%	2.6%	4.3%	1.5%
Total Failures (N)	89	109	39	151	184	136

categories do not fully capture the granular nature of the model's weaknesses, motivating our deeper clustering analysis.

4.2 Identifying Robust and Brittle Reasoning Modes

Our primary finding comes from the unsupervised clustering of all reasoning sentences across all model-dataset combinations. This analysis moves beyond error categorization to reveal a stark contrast between highly reliable reasoning modes and those that are exceptionally brittle. To formally validate our findings, we confirmed that the correctness rate for each selected cluster is statistically significant. We performed a Fisher's Exact Test, comparing each cluster's performance against a baseline sentence-level correctness rate derived from the overall problem-level accuracy for that specific model-dataset combination. For all highlighted clusters in Tables 2 and 3, the tests yielded p-values of less than 0.05, confirming that the observed performance is not due to random chance. There are other clusters that have a statistically significant difference in performance, but are omitted for brevity.

The results, presented in Tables 2 and 3 , showcase this performance gap. The models demonstrate near-perfect, statistically significant reliability in well-defined, procedural tasks. For example, on **GSM8K**, gpt-3.5-turbo-1106 achieved 100% correctness on clusters for *Calculating total cost of items*, and on **ASDiv**, gpt-4o-mini achieved 100% on *Calculating teams and student distribution*. This indicates a mastery of basic arithmetic and procedural execution.

In stark contrast, performance collapses when reasoning requires handling combinatorial constraints, algebraic abstraction, or complex multi-step logic. Most notably, several "Brittle" clusters exhibit a 0.0% correctness rate, such as gpt-3.5-turbo-1106 on **GSM8K** for *Calculating topping combinations with restrictions* and gpt-4o-mini on **ASDiv** for *Define variable and establish relationships*. This, along with other low-performing clusters, represents clear, systematic, and statistically significant failure modes.

Apart from insights into where models like gpt-3.5-turbo and gpt-4o-mini fail from a reasoning perspective, this work helps in understanding the difference between human and machine cognition. The cognitive profiles revealed here—exhibiting both absolute mastery and total, systematic failure on closely related mathematical concepts—are profoundly non-human-like. A human student might struggle with a concept, but they would not typically display this extreme binary of 100% success versus 0% failure. This discovery of "brittle" reasoning modes provides a more granular method for measuring the boundaries of an LLM's mathematical comprehension and offers a clear, data-driven roadmap for the targeted interventions needed to build more robust and reliable AI reasoners.

5 Limitations and Future Work

Our diagnostic framework relies on an LLM analyst (gpt-4o-mini), and a dedicated human validation of its categorization accuracy remains an important area for future investigation. While this camera-ready version has been expanded to include a comparative analysis across multiple models and datasets (Section 4.1, Table 1) and a multi-model cluster analysis (Section 4.2, Tables 2 and 3), we have also added further analysis to the appendix.This includes qualitative examples from robust and

Table 2: Robust and brittle reasoning clusters for gpt-4o-mini across datasets. Correctness shows the percentage of correct reasoning steps in each cluster. Count indicates the number of reasoning sentences. * indicates statistical significance (p < 0.05) compared to the overall baseline for that dataset.

Dataset	Category	Count	Correct %	Reasoning Mode
ASDiv	Brittle	17	0.0%	Define variable and establish relationships
	Brittle	12	0.0%	Equation manipulation and simplification
	Robust	10	100.0%*	Identifying and calculating quantities
	Robust	10	100.0%*	Calculating teams and student distribution
SVAMP	Brittle	20	15.0%*	Calculating total or average visitors
	Brittle	14	21.4%*	Identifying and adding quantities
	Robust	10	100.0%	Calculating totals and differences
	Robust	10	100.0%	Identifying and calculating Ferris wheel capacity

Table 3: Robust and brittle reasoning clusters for gpt-3.5-turbo-1106 across datasets. Correctness shows the percentage of correct reasoning steps in each cluster. Count indicates the number of reasoning sentences. * indicates statistical significance (p < 0.05) compared to the overall baseline for that dataset.

Dataset	Category	Count	Correct %	Reasoning Mode
GSM8K	Brittle	11	0.0%*	Calculating topping combinations with restrictions.
	Brittle	11	27.3%*	Calculate and round time or quantity.
	Robust	26	100.0%*	Calculating total cost of items.
	Robust	22	100.0%*	Sequential Calculation Steps.
ASDiv	Brittle	16	0.0%	Defining variables and calculating costs
	Brittle	15	0.0%	Establishing age relationships and equations
	Robust	10	100.0%*	Performing Subtraction to Find Remaining Amount
	Robust	10	100.0%*	Defining and solving for total distance
SVAMP	Brittle	10	0.0%*	Calculating totals through addition and subtraction
	Brittle	15	6.7%*	Calculating average and total visitors
	Robust	10	100.0%	Calculating total after additions
	Robust	10	100.0%	Calculating the difference between two numbers

brittle clusters (Appendix A) and a justification of the HDBSCAN clustering parameters (Appendix B) to address reviewer concerns.

To build on this work, our plan is twofold. First, we will expand our analysis to more complex reasoning domains, such as the MATH dataset. Second, and most critically, we plan to "close the loop" from diagnosis to solution. By using the sentences from identified brittle clusters (such as those in Table 3) as a targeted dataset for fine-tuning, we will investigate whether this data-efficient intervention can surgically repair these specific reasoning deficiencies—a key step toward building more robust and reliable models.

References

[1] Ricardo J. G. B. Campello, Davoud Moulavi, and Joerg Sander. Density-based clustering based on hierarchical density estimates. In *Advances in Knowledge Discovery and Data Mining*, pages 160–172. Springer Berlin Heidelberg, 2013.

[2] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems, 2021.

[3] Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step, 2023.

[4] Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. Self-refine: Iterative refinement with self-feedback, 2023.

[5] Shen-Yun Miao, Chao-Chun Liang, and Keh-Yih Su. A diverse corpus for evaluating and developing english math word problem solvers, 2021.

[6] Arkil Patel, Satwik Bhattamishra, and Navin Goyal. Are nlp models really able to solve simple math word problems?, 2021.

[7] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, pages 3982–3992, 2019.

[8] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models, 2022.

[9] Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models, 2023.

[10] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging llm-as-a-judge with mt-bench and chatbot arena, 2023.

A Qualitative Failure Analysis

To provide a more granular view of the failure modes identified in our high-level categorization (Section 4.1), we present two representative examples of incorrect reasoning traces. These examples illustrate the difference between a fundamental flaw in logic (Reasoning Error) and a flaw in execution (Calculation Error).

A.1 Example 1: Reasoning Error (gpt-3.5-turbo-1106 on SVAMP)

Question: *There are 17 different movies and 11 different books in the 'crazy silly school' series. If you read 13 of the books and watched 63 of the movies How many more movies than books are there in the 'crazy silly school' series?*

Model: gpt-3.5-turbo-1106

Error Type: Reasoning Error

Model's Reasoning Trace:

> *Step 1: First, we calculate the difference between the number of movies and the number of books: 63 movies - 13 books = 50 more movies than books.*

Analysis: The model correctly identified the required operation (subtraction) but failed to select the correct operands. It was distracted by the irrelevant quantities (63 movies watched, 13 books read) and used them instead of the total quantities given (17 movies, 11 books). This is a classic reasoning failure where the model's logical plan does not correctly map the question's constraints to the provided numbers. The correct calculation should have been $17 - 11 = 6$.

A.2 Example 2: Calculation Error (gpt-4o-mini on ASDiv)

Question: *For the final act, the circus brought out dancing animals wearing crowns. If each crown is made with 7 different colored feathers, how many feathers are needed for 934 crowns?*

Model: gpt-4o-mini

Error Type: Calculation Error

Model's Reasoning Trace:

Step 1: First, I identified that each crown requires 7 different colored feathers.

Step 2: Next, I calculated the total number of feathers needed by multiplying the number of crowns (934) by the number of feathers per crown (7).

*Step 3: The calculation is 934 crowns * 7 feathers/crown = 6558 feathers.*

Analysis: The model formulated a perfect logical plan. It correctly identified the operation (multiplication) and the correct operands (934 and 7). However, it failed at the final step of pure arithmetic. This demonstrates a calculation failure, as the model's reasoning was sound, but its execution was flawed. The correct calculation is $934 \times 7 = 6538$.

B Effects of Clustering Parameters

The cluster analysis in Section 4.2 relies on the HDBSCAN algorithm, which is sensitive to the ‘min cluster size‘ hyperparameter. This parameter controls the trade-off between cluster granularity and stability. We manually inspected the cluster outputs from a range of parameter settings.

We observed that low values (e.g., ‘min cluster size < 5‘) resulted in a high number of "micro-clusters" that were not semantically distinct or interpretable. Conversely, high values (e.g., ‘min cluster size > 10‘) grouped too many distinct reasoning patterns into single, overly broad clusters, masking the specific failure modes we sought to identify. The ‘min cluster size‘ of 6 used in this paper was selected as an optimal balance point, as it yielded the most interpretable and semantically coherent set of reasoning modes, clearly separating both robust and brittle categories.