Benchmarking LLMs on Advanced Mathematical Reasoning

Jonathan Yue
Daniel Klein

May 16, 2025

Benchmarking LLMs on Advanced Mathematical Reasoning

by Jonathan Yue

Research Project

Submitted to the Department of Electrical Engineering and Computer Sciences, University of California at Berkeley, in partial satisfaction of the requirements for the degree of **Master of Science, Plan II**.

Approval for the Report and Comprehensive Examination:

Committee:

Dan Klein

Professor Dan Klein
Research Advisor

05/15/2025

(Date)

* * * * * * *

Gireeja Ranade

Professor Gireeja Ranade
Second Reader

05/16/2025

(Date)

Benchmarking LLMs on Advanced Mathematical Reasoning

by

Jonathan Yue

A thesis submitted in partial satisfaction of the

requirements for the degree of

Master of Science

in

Electrical Engineering and Computer Sciences

in the

Graduate Division

of the

University of California, Berkeley

Spring 2025

Abstract

Benchmarking LLMs on Advanced Mathematical Reasoning

by

Jonathan Yue

Master of Science in Electrical Engineering and Computer Sciences

University of California, Berkeley

Professor Dan Klein, Chair

Large Language Models (LLMs) have improved dramatically at mathematical reasoning, progressing from basic arithmetic to olympiad level proofs. However, the existing, short-answer based benchmarks can suffer from limited scope for complex reasoning and therefore do not sufficiently measure the reasoning capabilities of LLMs. Formal proof-based benchmarks exist, but the need to convert problem statements into formal languages limits the scope of the problems. A potential reason for this significant gap in current literature is the difficulty in grading proof problems at scale. To address this, we first propose an LLM-as-a-judge framework to judge model-generated proofs and evaluated its efficacy. Then, we propose a benchmark of 77 PhD-level proof questions, drawn from Roman Vershynin's "High-Dimensional Probability: An Introduction with Applications in Data Science", and challenged state-of-the-art LLMs with these questions. We evaluated the LLM-generated solutions using the LLM-as-a-judge framework and found that, in general, state-of-the-art LLMs are still unable to adequately complete these proofs.

Contents

Acknowledgments

I am deeply grateful to my advisor, Dan Klein, for his support and mentorship over the past two years, and to Professor Gireeja Ranade for being my second reader and offering valuable conversations and feedback. I also wish to thank my mentor, Nicholas Tomlin, for his guidance and thoughtful input throughout this process.

I would like to thank all my classmates in the CompSci 294-271 course, AI in Education, taught by Professor Gireeja Ranade and Professor Narges Norouzi, where I engaged in discussions that inspired the conception of this project. I especially thank my team members in the course, Nora Guo and Sveinung Myhre, who I collaborated with to obtain the IRB approval and conduct a related study, for helping shape the ideas for this work.

Finally, I extend my heartfelt thanks to my family and friends for their constant encouragement and support, which has been invaluable in every facet of my life.

Chapter 1

Introduction

The advent of Large Language Models (LLMs) has resulted remarkable advancements in artificial intelligence, with mathematical reasoning emerging as a significant area of interest. LLMs have evolved from performing rudimentary arithmetic to assisting with mathematical discover (Romera-Paredes et al., 2024), driven by increased model size and sophisticated prompting techniques such as Chain-of-Thought (CoT) prompting (Wei et al., 2023) and Tool-Integrated Reasoning (TIR) (Ahn et al., 2024).

Despite these strides, a considerable gap persists in evaluating the deeper reasoning capabilities of LLMs. Existing benchmarks predominantly focus on short-answer or multiple-choice questions, such as MATH (Hendrycks et al., 2021) and GSM8K (Cobbe et al., 2021). While useful, these benchmarks often emphasize the final numerical output, potentially neglecting the rigor of the intermediate reasoning steps and suffering from issues such as benchmark saturation and data contamination (Hong et al., 2025; Petrov et al., 2025). Furthermore, their limited scope may not sufficiently assess complex conceptual understanding or the ability to construct intricate arguments (K. Huang et al., 2025). The poor performance of LLMs on problems that require rigorous proof generation, as opposed to numerical answers, suggests a potential "reasoning illusion," where success in some tasks might stem from pattern matching or tool assistance rather than genuine mathematical insight (Petrov et al., 2025).

On the other hand, benchmarks centered on formal proofs, such as Minif2f (K. Zheng et al., 2022) and PutnamBench (Tsoukalas et al., 2024), operate within the strict confines of symbolic systems like Lean or Isabelle. While valuable for assessing mechanically verifiable reasoning, the necessity of translating problem statements into these formal languages (autoformalization) can be a challenging task itself, which restricts the breadth of problems that can be addressed (Gulati et al., 2024; J. Zhang et al., 2025). This leaves a significant gap in evaluating the generation of novel, *natural language mathematical proofs*, which are more akin to human mathematical practice and crucial for interpretability. A primary obstacle to developing benchmarks for such proofs is the inherent difficulty in grading them consistently and at scale, as natural language proofs lack the immediate verifiability of formal proofs and often require expert human evaluation (Frieder et al., 2023).

To address these limitations and bridge the gap in evaluating advanced mathematical reasoning in natural language, this paper makes two primary contributions. First, we propose and evaluate an LLM-as-a-judge framework specifically designed for assessing the correctness and coherence of mathematical proofs. This approach offers a scalable alternative to manual expert evaluation. Second, leveraging this evaluation framework, we introduce a new benchmark comprising 77 PhD-level proof-based questions, designed to rigorously test the advanced reasoning capabilities of LLMs on problems requiring novel and intricate natural language proofs.

We find that, despite recent progress, current leading LLMs are generally still unable to adequately complete these complex proof-based tasks. This underscores the necessity for more challenging benchmarks that probe deeper levels of mathematical understanding and proof generation, thereby providing a more accurate measure of LLM reasoning abilities and guiding future research towards developing more capable and robust AI systems for mathematics.

Chapter 2

Related works

2.1 LLM for mathematical reasoning

The introduction of Large Language Models (LLMs) has marked a significant turn in artificial intelligence. Among the many applications, mathematical reasoning emerges as a key area of exploration and advancement. This section surveys the development of LLMs in tackling mathematical tasks.

Advances and Core Techniques in LLM Mathematical Reasoning

LLMs have rapidly improved at mathematical reasoning, progressing from simple arithmetic to solving more complex problems (Liu et al., 2025). A key driver for this progress is the increase in model size. Larger models follow instructions better and reason more effectively. Research shows that LLMs with over 100 billion parameters can solve difficult math problems with the right prompts (Forootani, 2025).

The discovery of several techniques have augmented this improvement due to model scale. A major cornerstone in this area is Chain-of-Thought (CoT) prompting. It guides LLMs to list steps in their reasoning before giving a final answer. This results in an improvement in both the performance on complex tasks and the interpretability of the reasoning process. Wei et al., 2023 showed that CoT prompting allows LLMs to deconstruct problems they would usually fail. Researchers have built upon basic CoT with more advanced versions and new types of "thinking" models. Examples include SoftCoT (Y. Xu et al., 2025), Continuous CoT (Cheng & Durme, 2024), and Coconut (Hao et al., 2024). SoftCoT, for instance, utilizes a smaller assistant model to generate "soft thought tokens" which are then projected into the primary LLM's representation space through a trainable module. This fine-tuning approach is parameter-efficient and has demonstrated enhanced performance on mathematical reasoning benchmarks.

The emergence of models explicitly designed for "thinking," such as OpenAI's O1 and

DeepSeek-R1, represents a notable progression. These models often take more time to reason during inference and are trained with Reinforcement Learning (RL) to build advanced cognitive skills like self-checking and reflection. DeepSeek-R1, for example, is reported to develop CoT reasoning capabilities autonomously through a pure RL training paradigm (DeepSeek-AI et al., 2025). Multi-round Thinking, where a model uses previous answers to try again and improve, has also led to better results, as seen in models like QwQ-32B and DeepSeek-R1 on difficult benchmarks such as AIME 2024 (Tian et al., 2025). RL-based reasoning methods are also being used in multimodal LLMs. Vision-R1, for instance, uses Progressive Thinking Suppression Training (PTST) to improve performance in solving multimodal math problems (W. Huang et al., 2025).

To address LLMs' weakness in precise numerical calculation and rigorous symbolic manipulation, Tool-Integrated Reasoning (TIR) has gained prominence (Ahn et al., 2024). TIR lets LLMs hand off subparts of a problem to tools like Python for math or symbolic solvers for algebra. This improves accuracy, especially when high precision or structured math is needed. For example, the winning solution in the AIMO-2 competition used CoT fine-tuning first, then fine-tuned again on a TIR dataset. This helped the model combine natural language reasoning with structured computation (Moshkov et al., 2025). TIR frameworks like TATA (Teaching LLMs According to Their Aptitude) (X. Xu et al., 2025) help models choose between CoT and TIR depending on the problem.

Still, LLMs struggle with highly advanced math problems, especially ones needing full proofs like those in math olympiads. A recent work evaluated state-of-the-art LLMs on 2025 USAMO problems. The results show significant limits (Petrov et al., 2025). Even a leading model like Gemini-2.5-PRO achieved an average score of only 25%, with other prominent models scoring below 5%. Common failure modes included flawed logical steps, introduction of unjustified assumptions, a lack of creative problem-solving strategies, and a tendency to falsely claim that a problem had been solved.

The observed discrepancy between LLM performance on benchmarks requiring numerical answers and those requiring rigorous proof generation suggests a potential "reasoning illusion." While techniques like CoT and TIR demonstrate improvements on certain types of mathematical problems, the poor performance on proof-based tasks suggests that these improvements might not always translate to genuine, deep mathematical understanding. LLMs may be leveraging pattern matching and tool-assisted computation to succeed in numerical answer-oriented tasks, without necessarily developing deep logical deductive capabilities. This illustrates the critical need for benchmarks that specifically evaluate these deeper reasoning and proof-generation abilities.

Formal vs. Informal Mathematical Reasoning by LLMs

Mathematical reasoning by LLMs can be broadly categorized into two domains: formal mathematical reasoning, which operates under the rigorous syntax of symbolic systems and proof assistants, and informal mathematical reasoning, which expresses mathematics in natural language.

Formal Mathematical Reasoning

This sub-field is characterized by its emphasis on mechanical verifiability and logical rigor.

LLMs for Formal Proof Generation (Lean, Isabelle, etc.): Much research is dedicated to training LLMs to generate proofs in formal languages such as Lean 4 (J. Zhang et al., 2025) and Isabelle (X. Zhao et al., 2024). The goal of this line of research is to develop systems that can autonomously produce machine-verifiable mathematical proofs.

Several notable models and systems have demonstrated progress. **AlphaGeometry** made headlines by solving Olympiad-level geometry problems. It combined a language model, which suggests potentially useful geometric constructions, with a symbolic deduction engine that formally verifies these steps. A key component of its development was the generation of 100 million synthetic data examples for training (Trinh et al., 2024). An improved version, **AlphaGeometry2** extended its formal language to handle a wider range of geometric problems and leveraged the Gemini architecture. AlphaGeometry2 reportedly surpassed the average performance of human gold medalists on a benchmark of IMO geometry problems (Chervonyi et al., 2025).

The **Self-play Theorem Prover (STP)** addresses the critical issue of data scarcity in formal mathematics by employing a dual-agent system: a conjecturer that proposes new theorems and a prover that attempts to prove them. This self-play loop allows the system to iteratively generate its own training data, which results in significant performance improvements on various reasoning benchmarks (Dong & Ma, 2025). **Goedel-Prover** is an open-source LLM designed for formal proof generation in Lean 4. It achieved state-of-the-art results by first formalizing a large dataset of natural language math problems into Lean 4 statements and then employing an expert iteration strategy to train the prover (Lin et al., 2025). Kimina-Prover Preview developed a reasoning-driven exploration paradigm, utilizing large-scale RL based on Qwen2.5-72B to generate proofs in Lean 4. This approach has also set new SOTA performance on the miniF2F benchmark (H. Wang et al., 2025).

Despite these advancements, a key challenge remains the scarcity of large-scale, high-quality datasets of formalized mathematical statements and proofs. The rigorous syntax of formal languages often makes proof generation in these systems more difficult for LLMs compared to natural language (J. Zhang et al., 2025).

Autoformalization: This critical sub-area focuses on the automatic translation of informal mathematical language into formal languages that can be processed by proof assistants. LLMs like GPT-4 have considerably advanced autoformalization capabilities through in-context learning. Techniques such as back-translation, where existing formal statements are automatically informalized into natural language to result in synthetic pairs of formal and informal mathematical statements, have been explored to augment training data (Yang et al., 2024).

However, autoformalization remains a difficult task, especially for complex or abstract mathematical concepts (Gulati et al., 2024). LLMs can struggle with selecting the correct preambles or library imports from vast formal mathematics libraries and may even generate references to non-existent preambles. Evaluating autoformalization faithfulness—that the formal statement accurately captures the semantics of the original informal input—remains an area of ongoing research.

Informal (Natural Language) Mathematical Proofs

This domain concerns the generation and comprehension of mathematical proofs, in the form most familiar to humans: in natural language.

Current Landscape: While CoT techniques enabled LLMs to generate natural language step-by-step reasoning, there has been relatively little research into the generation of rigorous and verifiable natural language mathematical proofs for complex problems. Much of the existing work on mathematical reasoning in LLMs focuses on solving problems that yield numerical or short-form answers.

The generation and evaluation of natural language proofs by LLMs present a unique set of challenges:

- **Verifiability**: A fundamental difficulty lies in the automated verification of natural language proofs. Unlike formal proofs, which can be mechanically checked by proof assistants, natural language proofs typically require human expert evaluation or highly sophisticated AI-driven verifiers. Even minor errors in logic or calculation can invalidate an entire proof, and these can be difficult for current systems to reliably detect in a natural language context.
- **Ambiguity**: Natural language is inherently ambiguous and often relies on implicit contextual knowledge and common sense. For instance, whether omitting certain steps in a mathematical argument is acceptable depends on the assumed level of familiarity of both the author and the audience. LLMs might not fully grasp these ambiguities and maintain the necessary level of precision throughout a proof. Another concern is that an LLM may inadvertently exploit these ambiguities and pretend as if it had completed a proof.
- **Data Scarcity for Complex Proofs**: High-quality, large-scale datasets of complex natural language proofs, particularly those with detailed step-by-step annotations or explanations of reasoning, are scarce. This is especially true for niche or advanced mathematical topics.

The comparatively smaller volume of research focused on generating and verifying natural language mathematical proofs by LLMs, as opposed to formal proofs or problem-solving with numerical answers, may be attributed to several factors. Formal systems and proof assistants make formal proof generation an attractive target for AI research as it provides a clear, objective measure of correctness. This reliable feedback loops for model training and evaluation stands in contrast to the inherent ambiguity of natural language. Additionally, formal mathematics libraries (e.g., Mathlib for Lean) offer a growing, structured, and verified corpus of mathematical knowledge. The field of Automated Theorem Proving (ATP) also has a long-standing tradition in AI, historically focusing on formal logic.

Despite these challenges, we believe that the generation and evaluation of complex informal mathematical proofs is vitally important. Natural language proofs are often more interpretable, which makes them more desirable in many settings. For instance, in an education setting, capabilities in complex mathematical reasoning may allow for more efficient proof grading or student hint generation. The space of mathematical reasoning can also act as a testbed to AI reasoning in less-defined real-world environments, where formalization would be nearly impossible and where interpretability would be ever more important.

2.2 Benchmarking Mathematical Reasoning in Large Language Models

The evaluation of mathematical reasoning capabilities in LLMs relies heavily on benchmarks. This section reviews existing benchmarks, categorizing them by their focus and problem types, and critically examines their limitations.

Short-Answer Benchmarks

These benchmarks typically assess LLMs on mathematical problems where the expected output is a final numerical answer or a concise textual response. Prominent examples include:

- **MathQA (Amini et al., 2019)**: Derived from the AQuA dataset, MathQA presents math word problems with multiple-choice answers. It was designed with an emphasis on interpretability by mapping problems to sequences of predefined mathematical operations (Amini et al., 2019).
- **MATH (Hendrycks et al., 2021b)**: This dataset is comprised of challenging problems from mathematics competitions such as the American Mathematics Competitions (AMC 10/12) and the American Invitational Mathematics Examination (AIME). While solutions often require multiple reasoning steps, evaluation typically focuses on the correctness of the final numerical answer (Hendrycks et al., 2021).
- **GSM8K (Cobbe et al., 2021)**: Consisting of grade school mathematics word problems, GSM8K is designed to test multi-step arithmetic reasoning. Models are expected

to produce a numerical answer derived through a sequence of calculations (Cobbe et al., 2021).

Despite their widespread use, these short-answer benchmarks face several criticisms:

- **Emphasis on Final Answers**: A primary limitation is their predominant focus on the correctness of the final answer, often neglecting the intermediate reasoning steps. This evaluation approach can inadvertently reward models that arrive at the correct solution through flawed reasoning processes (Petrov et al., 2025).
- **Benchmark Saturation**: Many of these benchmarks are rapidly approaching saturation, with state-of-the-art LLMs achieving very high accuracy scores (e.g., over 97% on GSM8K as reported in Budagam et al., 2024). This saturation diminishes their efficacy in differentiating the capabilities of advanced models.
- **Data Contamination and Overfitting**: A significant concern is the potential for benchmark data to have been included in the vast training corpora of LLMs. This data contamination can lead to inflated performance metrics that reflect memorization rather than true reasoning ability. Studies that have created perturbed versions of these benchmarks, such as GSM1k (a contamination-free version of GSM8K) (H. Zhang et al., 2024) and RV-Bench (which introduces random variables into MATH problems) (Hong et al., 2025), have often observed notable performance drops, suggesting that models may be overfitting to the original benchmark distributions.
- **Limited Scope for Complex Reasoning**: These benchmarks, often centered on arithmetic or basic algebraic manipulation, may not adequately test deeper conceptual understanding, abstract reasoning, or the ability to construct complex proofs (K. Huang et al., 2025). MathQA, for example, explicitly omits problems involving higher-order polynomials or entirely non-numeric solutions.
- **Sensitivity to Input Perturbations**: LLM performance on these benchmarks can be surprisingly fragile, exhibiting significant degradation in response to minor alterations in problem phrasing, numerical values, or the introduction of irrelevant information. This sensitivity suggests a lack of robust understanding of underlying mathematical principles. For instance, MATH-P-Hard introduces hard perturbations to MATH problems so that the original solution steps don't apply, resulting in significant performance drops. The researchers found that the models blindly apply learned problem-solving tactics even towards unsuitable problems (K. Huang et al., 2025).
- **Lack of Diversity in Problem Types and Assessed Reasoning Skills**: Existing benchmarks may not encompass a sufficiently broad range of mathematical topics or the diverse cognitive skills required for advanced mathematical thought. The MaTT benchmark, for example, found that LLM performance can vary significantly even

across closely related subtopics within the same general mathematical area (Davoodi et al., 2025).

Olympiad-Level and Proof-Focused Benchmarks

To address the limitations of short-answer benchmarks and to probe more advanced mathematical reasoning, several benchmarks focusing on Olympiad-level problems and proof generation have been developed.

- **Minif2f (K. Zheng et al., 2022)**: This benchmark provides a collection of 488 formal problem statements derived from Olympiad-level mathematics (AIME, AMC, IMO) and undergraduate courses, formalized in multiple interactive theorem proving systems like Lean, Isabelle, and HOL Light. It primarily covers algebra, number theory, and inequalities, with formalizations done manually.
- **PutnamBench (Tsoukalas et al., 2024)**: Sourced from the challenging William Lowell Putnam Mathematical Competition, PutnamBench features 1692 hand-constructed formalizations of 640 theorems in Lean 4, Isabelle, and Coq. Current models have demonstrated very limited success on this benchmark, solving only a handful of the problems.
- **Omni-math (Gao et al., 2025)**: This benchmark aims to be a universal Olympiad-level mathematics evaluation suite, comprising 4,428 problems spanning 33 sub-domains and 10 difficulty levels. It utilizes GPT-4o as a judge (Omni-Judge) for evaluating answers, which are typically numerical or SymPy objects.
- **FrontierMath (Glazer et al., 2024)**: This benchmark introduces hundreds of original, exceptionally challenging mathematics problems at the advanced undergraduate, graduate, and research levels, crafted and vetted by expert mathematicians. It covers a broad range of modern mathematics and employs automated verification for computable answers (often large integers or SymPy objects). The problems are novel and not released, so there is minimal risk of data contamination. State-of-the-art LLMs currently solve fewer than 2% of these problems .
- **FormalMATH (Yu et al., 2025)**: A large-scale benchmark consisting of 5,560 formally verified mathematical statements in Lean 4, significantly larger than miniF2F. Evaluations of LLM-based theorem provers on FormalMATH revealed low success rates (e.g., Kimina-Prover achieved 16.46% pass@32) and pronounced domain bias, with models performing better in algebra but struggling in areas like calculus.

These advanced benchmarks, while pushing the boundaries of difficulty, still tend to focus on problems with verifiable numerical or symbolic answers (Omni-math, FrontierMath) or on proofs within formal systems (Minif2f, Putnam Bench, FormalMath). The benchmarks focusing on formal proofs also have a relatively narrow scope, focusing on algebra and number theory, with less coverage of problem types that are harder to express in certain formal systems, such as geometry or probability. Thus, these benchmarks may not fully represent the breadth of mathematical topics. This leaves a gap in evaluating the generation of *novel natural language proofs*, particularly in sub-domains not emphasized by these benchmarks.

2.3 LLM-as-a-Judge for Evaluating Mathematical Reasoning

The paradigm of using Large Language Models as evaluators, commonly termed "LLM-as-a-judge," has gained considerable traction as a scalable alternative to human assessment. While research in LLM-as-a-judge techniques has previously focused on natural language generation tasks such as text summarization, we hope that this framework can be generalized to natural language mathematical arguments. This section delves into the methodologies behind LLM-as-a-judge systems, their application to evaluating mathematical reasoning and proofs, and the associated challenges of performance, reliability, and inherent biases.

Methodologies and Applications of LLM-as-a-Judge

The fundamental concept of LLM-as-a-judge involves employing capable LLMs, often large, general-purpose frontier models like GPT-4, to assess the quality of outputs generated by other LLMs or even human-produced text. The primary motivation is to approximate human preferences and judgments at scale, thereby reducing the time and cost associated with manual annotation and evaluation.

Several common methodologies are employed in LLM-as-a-judge systems (Li et al., 2024):

- Pointwise Scoring: This approach involves the LLM judge assigning a direct numerical score to a single response. The score can be on a Likert scale (e.g., 1 to 5) or a continuous scale, reflecting the perceived quality of the response based on predefined criteria or an overall assessment.
- Pairwise Comparison/Ranking: In this setup, the LLM judge is presented with two (or more) responses to the same prompt or problem and is tasked with determining which response is superior or providing a relative ranking. This method is frequently used in benchmarks like MT-Bench and Chatbot Arena for evaluating conversational AI.
- Listwise Judgment: This extends pairwise comparison to scenarios where the LLM judge evaluates and ranks a list of multiple responses simultaneously.

The LLM-as-a-judge paradigm was initially prominent in the evaluation of general Natural Language Generation (NLG) tasks such as text summarization, machine translation, and dialogue systems. However, it is increasingly being explored for more structured and specialized domains. For instance, Stephan et al., 2025 have investigated the application of LLM judges to mathematical problems in a pair-wise comparison fashion.

Metrics for Judge Performance

- **Correlation with Human Experts**: A primary indicator of an LLM judge's utility is the degree to which its evaluations align with those provided by human experts in the domain. High correlation coefficients are often cited as evidence of the LLM judge's ability to capture human-like assessment criteria (Gu et al., 2025).
- **Consistency**: An ideal LLM judge should produce consistent evaluations for semantically similar inputs or when irrelevant factors, such as the order of presentation in pairwise comparisons, are altered. However, achieving high consistency is often challenging due to inherent biases.
- **Accuracy**: In domains like mathematics, where the correctness of a short-answer question can be objectively determined, the accuracy of the LLM judge's assessment (e.g., correctly identifying a proof as valid or invalid) becomes a key metric.

Known Biases

LLM judges are not infallible and have been shown to exhibit several systematic biases that can distort their evaluations:

- **Position Bias**: LLM judges may favor responses presented in a particular position, most commonly the first response in a pairwise comparison, irrespective of content quality. L. Zheng et al., 2023 highlighted this as a significant issue, finding that GPT-4 exhibited a preference for the first presented answer in over 60% of cases when evaluating similar responses .
- **Verbosity Bias**: LLM judges may show a preference for longer, more verbose responses, even if these responses are not necessarily more accurate, clear, or substantively better than shorter alternatives (L. Zheng et al., 2023).
- **Self-Preference/Self-Enhancement Bias**: LLMs can exhibit a bias towards outputs generated by themselves or by models from the same family or with a similar architectural style (L. Zheng et al., 2023). Research suggests this bias might be linked to the LLM judge preferring texts that have lower perplexity under its own language model—that is, texts that are more familiar or predictable to it (Wataoka et al., 2024).
- **Style/Superficial Reflection Bias**: LLM judges can be unintentionally influenced by the writing style, fluency, or other superficial characteristics of a response, rather

than its core correctness and logical rigor. A specific form of this, superficial reflection bias, has been identified where reasoning-focused LLMs are swayed by phrases that merely mimic reasoning (e.g., "let me think...") without necessarily reflecting sound logic (Q. Wang et al., 2025).

Mitigating these biases is crucial for developing reliable LLM-as-a-judge systems.

LLM-as-a-Judge in the Context of Mathematical Proofs

The use of LLMs in automated grading and feedback systems for mathematical proofs is an emerging area. Systems for autograding specific types of proofs, such as mathematical induction proofs, have shown promise (C. Zhao et al., 2025). **MATH-Minos** is a system that aims to improve mathematical verifiers (critics) by training them with stepwise natural language feedback (Gao et al., 2024). This approach helps the verifier learn to identify various types of errors, including logical flaws, more effectively than training with binary correct/incorrect labels alone.

There is also progress made on LLM self-verification and self-correction in mathematical reasoning. **SelfCheck (Miao et al., 2023)** uses a zero-shot method to enable an LLM to check its own step-by-step reasoning. It prompts the model to regenerate each step based on the preceding ones and then compare the regenerated step with the original. It has demonstrated some effectiveness in improving final answer accuracy on math word problem benchmarks like GSM8K, MathQA, and MATH (Miao et al., 2023). The **Natural Program** framework proposed by Ling et al., 2023 uses a natural language-based deductive reasoning format that structures CoT outputs to facilitate step-by-step self-verification by the LLM. Each step explicitly cites its premises, allowing for focused verification. While this can improve the rigor of the reasoning chain, its effectiveness can be hampered by the inherent ambiguities of natural language.

Applying the LLM-as-a-judge paradigm to the evaluation of mathematical proofs introduces specific challenges due to the nature of mathematical argumentation. Mathematical proofs demand strict logical coherence and deductive validity. LLM judges must therefore go beyond surface-level textual understanding to assess whether each step in a proof is logically sound. This is a significantly more demanding task than evaluating general text quality or stylistic attributes. A particularly subtle challenge is dealing with novelty of a proof strategy. An LLM judge, potentially trained on a vast corpus of existing mathematical texts, might favor standard, familiar proof techniques over novel, creative, yet correct approaches, simply because the latter are less represented in its training data.

Additionally, there is evidence suggesting a strong relationship between an LLM's ability to solve mathematical problems and its capacity to judge solutions to those problems. Krumdick et al., 2025 found that an LLM judge's performance in assessing correctness is significantly impacted by its own ability to answer the question at hand: judges tend to struggle with evaluating responses to questions they themselves cannot solve accurately.

Chapter 3

Data collection

Aiming to ultimately measure state-of-the-art models' mathematical reasoning capabilities, we use two sources of data in this work, corresponding to the two main components of this work. First, to evaluate the efficacy of our proposed LLM-as-a-judge framework, we use a data set of 100 student proof attempts and the corresponding course staff evaluations on a pre-defined rubric. Secondly, to benchmark the capacity of state-of-the-art LLMs in producing advanced mathematical reasoning, we sourced a publicly available graduate-level math textbook and a publicly available set of solutions to some of the exercise problems in the textbook.

3.1 Ground truth human proof evaluation

To measure the efficacy of our LLM-as-a-judge framework, we need a source of ground truth proof evaluation developed by human experts. This way, we can compare the evaluation done by humans and by the LLM.

The data for this study came from student responses (and the corresponding course staff evaluations) to proof-based questions in final exam of the fall 2018 session of EECS 16A, an undergraduate introductory linear algebra course at the University of California, Berkeley, taught by Professor Vladimir Stojanovic and Professor Gireeja Ranade. The exam questions and evaluations are credited to the professors and course staff members.

The student answers were collected in accordance with the IRB approval we received and are fully anonymized. A total of 100 student answers were collected. The digital scans of the students' responses were converted into LaTeX format to ensure a standardized and analyzable dataset.

In addition to the student-generated content, the dataset includes evaluations of each student's answer conducted by the EECS 16A course staff. These evaluations were performed using a 4-point scoring rubric designed to assess the correctness of the proofs. The rubric categories were defined as follows:

- +4 points (Completely correct): The proof was logically sound, well-structured, and demonstrated a complete understanding of the concepts, with no mathematical errors.
- +3 points (Correct but small error): The proof was largely correct and demonstrated a good understanding, but contained minor inaccuracies or omissions that did not fundamentally undermine the overall argument (e.g., a computational slip, a missing minor condition).
- +2 points (Major error): The proof exhibited significant flaws, such as a misunderstanding of key definitions or theorems, or a critical logical gap.
- +0 points (Incorrect/Blank): The response was entirely incorrect, irrelevant to the question asked, or left blank.

This collection process yielded a rich dataset comprising the students' original proof attempts and corresponding expert assessments of their quality.

3.2 Mathematical text and ground truth solutions

This study uses the content and exercises in the first edition of Roman Vershynin's "High-Dimensional Probability: An Introduction with Applications in Data Science" (Vershynin, 2018) as a base for a data set in advanced mathematical reasoning. This textbook is publicly available. [1] It was selected because it contained difficult concepts that required deep and nuanced mathematical understanding. Its introduction describes its intended audience as "doctoral and advanced masters students and beginning researchers in mathematics, statistics, electrical engineering, computational biology and related areas" and describes a prerequisite of "a rigorous course in probability theory (on Masters or Ph.D. level)".

The focus of the book, probability, means that the book does not contain many figures. This is ideal because we are not directly interested in models' multi-model capabilities. Probability is also an area where it is difficult to convert into formal math language, so it is under-represented in formal math data sets.

Additionally, the book has a gradient in difficulty. The initial chapters introduce foundational concepts at a relatively accessible level (e.g. classical inequalities in probability), while the complexity and specificity of topics in later chapters present challenges that are not readily solvable by current state-of-the-art LLMs.

For the purpose of establishing verifiable ground truth solutions to exercises from this textbook, we draw upon a publicly accessible GitHub repository. [2] This repository contains solutions developed by Pingbang Hu, a Ph.D. candidate at University of Illinois Urbana-Champaign. These solutions are treated as the benchmark against which problem-solving

[1] https://www.math.uci.edu/ rvershyn/papers/HDP-book/HDP-book.html
[2] https://github.com/sleepymalc/HDP-Solution

attempts are compared. We have done some cursory cross-checking of these solutions to confirm the accuracy of these solutions.

The solutions already come in LaTeX format, and the textbook is converted into LaTex through OCR methods. We run experiments using both of these sources but will not redistribute them.

In total, **77** exercise questions were chosen across chapters 0 through 6 of the book, covering topics such as concentration inequalities, high-dimensional random vectors and matrices, and quadratic forms. Vershynin also helpfully indicated the difficulty of these exercises, ranging from trivial (1) to challenging (4). A summary statistic of the questions is as follows:

chapter	**num_questions**	**avg_difficulty**
0	2	2.0
1	4	1.0
2	23	1.91
3	17	2.24
4	15	1.8
5	7	2.86
6	9	1.89

An example [question, ground truth answer] tuple is shown in Figure 5.1.

Combined, the textbook and solution provide an ideal test set to evaluate models' ability to construct novel mathematical arguments and reason in unfamiliar domains.

Chapter 4

LLM-as-a-judge for mathematical proofs

In this chapter, we aim to develop and evaluate an LLM-as-a-judge framework for assessing the correctness and quality of natural language mathematical proofs. We approach the development iteratively, exploring different ways of incorporating rubrics to guide the LLM's evaluation process. Our goal is to understand how different rubric strategies impact the LLM's ability to align with human expert judgment. We experiment with the following three approaches of evaluating proofs using the data in Section 3.1.

- **Baseline: LLM-as-a-judge with no rubric**

  In this initial approach, we establish a baseline for LLM performance without a rubric. The LLM is tasked with evaluating student-generated proofs on a 4-point scale, where 4 represents a fully correct proof and 0 a fully incorrect proof. No additional instruction is given about how the model should give out partial credit. This method assesses the model's ability to directly quantify the goodness of a solution.

- **LLM-as-a-judge with ground truth rubric**

  This iteration leverages the detailed, pre-existing rubric that was used by the course staff to grade the student answers. This ground truth rubric, as described in our data collection section, consists of a 4-point scale with specific descriptors for what constitutes a completely correct proof, a proof with small errors, a proof with major errors, or an incorrect/blank response. The LLM is provided with the student's proof and this exact course-staff rubric. Its task is to assign a score to the proof according to the given rubric's criteria. This setup aims to test how well the LLM can internalize and apply a detailed, externally validated scoring guide, with the hopes of producing evaluations that are more closely aligned with the original human graders' intentions. The primary challenge here lies in the LLM's ability to accurately interpret the nuanced language of the rubric and consistently apply it to diverse student responses.

- **LLM-as-a-judge with automatic rubric generation as an intermediary step**

  In many real-world scenarios, a detailed, validated rubric like the one provided by course staff may not be available. To address this, we explore a two-stage process where an LLM first automatically generates a rubric, which is then used by an LLM (could be the same or a different model) to judge the student proofs. The motivation is to create a more scalable and adaptable framework that does not rely on pre-existing, human-annotated rubrics for every new problem.

  For the rubric generation phase, we prompt an LLM as if it were a course staff member tasked with creating a grading scheme. The LLM is provided with the specific proof problem and the corresponding staff-provided model answer. It is then instructed to identify the most critical logical steps in the solution. A sample generated rubric can be seen in Figure 4.1. As seen in the figure, with appropriate prompting, LLMs are capable of identifying the key steps used in the proof and create a detailed rubric. They are also, to a certain extent, able to understand the nuances of the natural language proof. For instance, it understands that the staff solution derived $\vec{v}_\ell^T \mathbf{Q} \vec{v}_k = \vec{v}_\ell^T \mathbf{A}^T \mathbf{A} \vec{v}_k = \vec{v}_\ell^T \lambda_k \vec{v}_k$ from using the eigenvector equation, which is not explicitly stated in the solution. The model's ability to infer the intended logical flow from the staff solution is persistent, even when tasked with more difficult proofs.

  The quality of the subsequent proof evaluation in this framework heavily depends on the relevance and accuracy of the auto-generated rubric. This approach introduces an additional layer of complexity but offers the potential for a highly automated assessment pipeline.

Question: Given two distinct eigenvalue/eigenvector pairs, $(\lambda_k, \vec{v}_k)$ and $(\lambda_\ell, \vec{v}_\ell)$, show that for the symmetric matrix $\mathbf{Q} = \mathbf{A}^T\mathbf{A}$, if $\lambda_k \neq \lambda_\ell$ then $\langle \vec{v}_k, \vec{v}_\ell \rangle = 0$ (i.e. any pair of eigenvectors with distinct eigenvalues is orthogonal). Hint: $\mathbf{Q}\vec{v}_k = \lambda_k \vec{v}_k$ and $\vec{v}_\ell^T \mathbf{Q} = \lambda_\ell \vec{v}_\ell^T$.

Solution: Consider	**Rubric:** max score: 4
$\vec{v}_\ell^T \mathbf{Q}\vec{v}_k = \vec{v}_\ell^T \mathbf{A}^T\mathbf{A}\vec{v}_k = \vec{v}_\ell^T \lambda_k \vec{v}_k$	+1: Use the right eigenvalue equation $\mathbf{Q}\vec{v}_k = \lambda_k\vec{v}_k$ to compute $\vec{v}_\ell^T\mathbf{Q}\vec{v}_k = \lambda_k\vec{v}_\ell^T\vec{v}_k$. This step demonstrates the ability to apply the given eigenvalue equation for the eigenvector $\vec{v}_k$ and manipulate the expression to isolate the dot product, showing understanding of how eigenvalues relate to the matrix action.
$\vec{v}_\ell^T \mathbf{Q}\vec{v}_k = \vec{v}_\ell^T \mathbf{A}^T\mathbf{A}\vec{v}_k = \lambda_\ell \vec{v}_\ell^T \vec{v}_k$	+1: Use the left eigenvalue equation $\vec{v}_\ell^T\mathbf{Q} = \lambda_\ell\vec{v}_\ell^T$ to compute $\vec{v}_\ell^T\mathbf{Q}\vec{v}_k = \lambda_\ell\vec{v}_\ell^T\vec{v}_k$. This step applies the left eigenvalue equation for $\vec{v}_\ell$, highlighting the symmetry properties or given conditions of $\mathbf{Q}$, and reinforces the connection between left eigenvectors and the matrix transpose in symmetric matrices.
We see the two expressions on the left are equal, so set the two expressions on the right equal to each other: $$\lambda_k \vec{v}_\ell^T \vec{v}_k = \lambda_\ell \vec{v}_\ell^T \vec{v}_k$$	+1: Set the two expressions for $\vec{v}_\ell^T\mathbf{Q}\vec{v}_k$ equal, yielding $\lambda_k\vec{v}_\ell^T\vec{v}_k = \lambda_\ell\vec{v}_\ell^T\vec{v}_k$. This step involves recognizing that both derivations represent the same scalar quantity, allowing the formation of an equation that links the eigenvalues and the dot product, which is crucial for subsequent algebraic manipulation.
If $\lambda_k \neq \lambda_\ell$, then the only possible solution is that $\vec{v}_\ell^T\vec{v}_k = 0$, which means $\vec{v}_\ell$ and $\vec{v}_k$ are orthogonal.	+1: Given $\lambda_k \neq \lambda_\ell$, conclude that $\vec{v}_\ell^T\vec{v}_k = 0$, implying orthogonality. This step requires solving the equation algebraically, noting that unequal coefficients imply the dot product must be zero, and demonstrates the logical inference from the eigenvalue difference to vector orthogonality in the context of symmetric matrices.

Figure 4.1: Sample (Question, Answer, Rubric) tuple for evaluating LLM-as-a-judge. Correspondence between the solution and LLM-generated rubric is shown in color

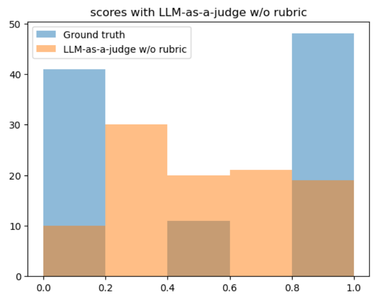
4.1 Evaluation

We evaluate each of the aforementioned LLM-as-a-judge frameworks using the dataset of 100 student answers from the undergraduate linear algebra course. The primary metric for assessing performance is the Pearson correlation coefficient between the scores assigned by the LLM framework and the ground truth scores provided by the course staff. Higher correlation values generally indicate better agreement between the LLM's judgment and human expert evaluation (Gu et al., 2025). We generated the rubric and all evaluations using Grok 3 mini with reasoning (xAI, 2025). This model was chosen because it is one of the state-of-the-art models on general reasoning tasks, which helps with understanding the nuances of the staff solution, as well as the generation and evaluation of mathematical arguments.

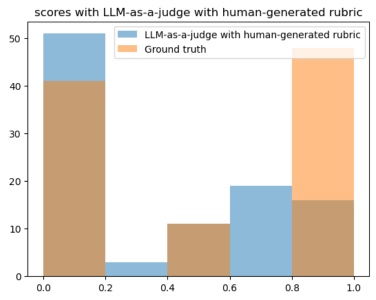
The quantitative results of our evaluation, specifically the Pearson correlation coefficients, are summarized in Table 4.1. Visualizations of the LLM scores versus human scores for different rubric configurations are presented in Figure 4.2.

LLM-as-a-judge Framework	**Pearson Correlation (ρ)**
Baseline: LLM with manually generated rubric	0.786
LLM with human-generated rubric	0.874
LLM with LLM-generated rubric	0.857

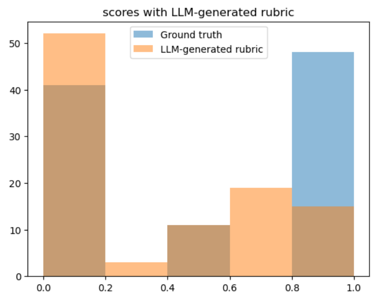
Table 4.1: Correlation of LLM-as-a-judge frameworks with ground truth scores.



(a) Baseline (no rubric)



(b) Human-generated rubric



(c) LLM-generated rubric

Figure 4.2: The distribution of LLM-as-a-judge score vs. ground truth score.

As seen from these results, LLM-as-a-judge performs best with the course staff created rubric, achieving a correlation of 0.874. It performs the worst when no rubric was provided at all. We see that the LLM-generated rubric performs nearly as well as the course staff created one, suggesting that it can act as an effective proxy for the true rubric in settings where that is not available. Due to this result, in the following sections, all evaluation on LLM-generated math proofs will first generate an intermediate rubric, then grade the solution in accordance with that rubric.

We now present an example of when the LLM-generated rubric helped with alignment with human experts.

Question: Same as in Figure 4.1

Student solution: [Ground truth score: 0/4] We have that $\mathbf{Q}\vec{v_k} = \lambda_k \vec{v_k}$ and $\vec{v_\ell}^T \mathbf{Q} = \lambda_\ell \vec{v_\ell}^T$. Also, $\mathbf{Q} = \mathbf{A}^T\mathbf{A} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^{-1}$ where $\mathbf{V}$ is the matrix of eigenvectors as columns and $\mathbf{\Lambda}$ is the matrix of λ as diagonals. Because $\mathbf{Q}$ is symmetric, $\mathbf{V}$ is an orthonormal matrix, which means that each of the columns of $\mathbf{V}$ is orthogonal to the others. The columns of $\mathbf{V}$ are $\vec{v_k}$ and $\vec{v_\ell}$ (there could be more). Since $\vec{v_k}$ and $\vec{v_\ell}$ are orthogonal, $\langle \vec{v_k}, \vec{v_\ell} \rangle = 0$ because $\langle \vec{v_k}, \vec{v_\ell} \rangle = \|\vec{v_k}\|\|\vec{v_\ell}\| \cdot \cos(90°)$ and $\cos(90°) = 0$.

Evaluation without rubric:
The student correctly identify that $\mathbf{Q} = \mathbf{A}^T\mathbf{A}$ is symmetric and use the property that symmetric matrices have orthonormal eigenvectors. Additionally, they provide an informal explanation using the cosine of 90 degrees, which, while not necessary, shows some intuition.
However, there are several issues that prevent a perfect score. The student did not use the provided hint, which suggested manipulating the equations $\mathbf{Q}\vec{v_k} = \lambda_k \vec{v_k}$ and $\vec{v_\ell}^T \mathbf{Q} = \lambda_\ell \vec{v_\ell}^T$ to directly show orthogonality. Instead, they relied on the spectral theorem and diagonalization, which is valid but bypasses the intended derivation and may not demonstrate the same level of engagement with the specific matrix operations.
Despite these shortcomings, the student's reasoning is logically sound and arrives at the correct conclusion. This warrants a score deduction for not adhering to the hint, but the answer is still largely correct.
Final score: 3/4

Evaluation with LLM rubric:
+0: Use the right eigenvalue equation $\mathbf{Q}\vec{v_k} = \lambda_k \vec{v_k}$ to compute $\vec{v_\ell}^T \mathbf{Q}\vec{v_k} = \lambda_k \vec{v_\ell}^T \vec{v_k}$. The student mentioned the eigenvalue equation for $\vec{v_k}$ but did not manipulate it to isolate the dot product as required.
+0: Use the left eigenvalue equation $\vec{v_\ell}^T \mathbf{Q} = \lambda_\ell \vec{v_\ell}^T$ to compute $\vec{v_\ell}^T \mathbf{Q}\vec{v_k} = \lambda_\ell \vec{v_\ell}^T \vec{v_k}$. The student wrote an incorrect expression $\vec{v_\ell}^T Q \vec{v_\ell} = \lambda_\ell \vec{V_\ell}^T$, which does not properly use the left eigenvalue equation to compute $\vec{v_\ell}^T \mathbf{Q}\vec{v_k}$.
+0: Set the two expressions for $\vec{v_\ell}^T \mathbf{Q}\vec{v_k}$ equal, yielding $\lambda_k \vec{v_\ell}^T \vec{v_k} = \lambda_\ell \vec{v_\ell}^T \vec{v_k}$. The student did not compute or set equal any expressions for $\vec{v_\ell}^T \mathbf{Q}\vec{v_k}$.
+0: Given $\lambda_k \neq \lambda_\ell$, conclude that $\vec{v_\ell}^T \vec{v_k} = 0$, implying orthogonality. The student concluded orthogonality but based it on the orthonormality of the eigenvector matrix due to Q being symmetric, not on solving the equation $\lambda_k \vec{v_\ell}^T \vec{v_k} = \lambda_\ell \vec{v_\ell}^T \vec{v_k}$ algebraically as specified.
Final score: 0/4

Figure 4.3: LLM-generated rubric successfully steered evaluation towards alignment with the correct score. The student solution exhibited circular logic, assuming that a symmetric matrix's eigenvectors are orthonormal. The evaluation without rubric was unable to identify this mistake. Additionally, it was also distracted by the student's use of cosine. It acknowledged that it as an unnecessary step but counted it in favor for the student ("shows some intuition"). Evaluation with LLM-generated rubric correctly judged this proof.

Chapter 5

LLM natural language proof-writing

In this chapter, we shift our focus from evaluating student-generated proofs to assessing the capabilities of Large Language Models (LLMs) in generating natural language mathematical proofs for complex, previously unseen problems. To achieve this, we leverage the LLM-as-a-judge framework, developed and refined in the previous chapter, as our primary evaluation tool. This allows for a systematic and rubric-based assessment of the LLM-generated proofs, providing nuanced insights into their correctness, logical coherence, and completeness. The problems are sourced from an advanced textbook, ensuring a high degree of difficulty.

5.1 Methodology

Our methodology for evaluating LLM proof-writing capabilities centers around presenting LLMs with challenging proof problems and then using our LLM-as-a-judge framework to score the generated solutions. The core components of this process are the problem instances, the inclusion of contextual information, and the evaluation procedure.

Problem Instances: (Question, Staff Answer, Rubric) Tuples

Each problem instance in our evaluation set is defined by a tuple consisting of:

1. **The Question**: The mathematical proof problem statement as presented in the source.
2. **The Staff Answer (Ground Truth Solution)**: A detailed, correct solution to the problem, typically mirroring a model solution one might expect from an expert or find in a solutions manual. This serves as the gold standard.
3. **The Rubric (Ground Truth Rubric)**: A detailed, points-based rubric specifically designed for the question, outlining key logical steps and criteria for evaluating a proof. This rubric is used by the LLM-as-a-judge framework to score the LLM-generated proofs.

Below is an example of such a tuple, derived from Exercise 0.0.5 of Roman Vershynin's "High-Dimensional Probability."

Question: Prove the inequalities

$$\left(\frac{n}{m}\right)^m \leq \binom{n}{m} \leq \sum_{k=0}^{m} \binom{n}{k} \leq \left(\frac{en}{m}\right)^m$$

for all integers $m \in [1, n]$.

Solution:
Fix some $m \in [1, n]$. We first show $(n/m)^m \leq \binom{n}{m}$. This is because
$\frac{(n/m)^m}{\binom{n}{m}} = \prod_{j=0}^{m-1} \left(\frac{n}{m}\frac{m-j}{n-j}\right)$
≤ 1
as $\frac{n-j}{m-j} \geq \frac{n}{m}$ for all j.

The second inequality $\binom{n}{m} \leq \sum_{k=0}^{m} \binom{n}{k}$ is trivial since $\binom{n}{k} \geq 1$ for all k.
The last inequality is due to
$\frac{\sum_{k=0}^{m}\binom{n}{k}}{\left(\frac{n}{m}\right)^m} \leq \sum_{k=0}^{n} \binom{n}{k} \left(\frac{m}{n}\right)^k$

$= \left(1 + \frac{m}{n}\right)^n$

$\leq e^m$.

Rubric: max score: 8
+1: Fix an integer m with $1 \leq m \leq n$.
+1: Express $\frac{\left(\frac{n}{m}\right)^m}{\binom{n}{m}} = \prod_{j=0}^{m-1} \frac{n}{m}\frac{m-j}{n-j}$, using the definition of the binomial coefficient.
+1: Prove that for all j, $\frac{n}{m}\frac{m-j}{n-j} \leq 1$ by showing $\frac{n-j}{m-j} \geq \frac{n}{m}$, which holds because $\frac{n-j}{m-j} - \frac{n}{m} = \frac{(n-m)j}{m(m-j)} \geq 0$ given $n \geq m$ and $m - j > 0$.
+1: Conclude that the product $\prod_{j=0}^{m-1} \frac{n}{m}\frac{m-j}{n-j} \leq 1$, implying $\left(\frac{n}{m}\right)^m \leq \binom{n}{m}$, since all factors are positive and less than or equal to 1.
+1: Note that $\binom{n}{m} \leq \sum_{k=0}^{m} \binom{n}{k}$, as the sum includes $\binom{n}{m}$ and other non-negative binomial coefficients $\binom{n}{k} \geq 0$ for $k = 0$ to m.
+1: Consider the ratio $\frac{\sum_{k=0}^{m}\binom{n}{k}}{\left(\frac{n}{m}\right)^m}$ and bound it above by $\sum_{k=0}^{n} \binom{n}{k} \left(\frac{m}{n}\right)^k$, by observing that for $k \leq m$, $\binom{n}{k}\left(\frac{m}{n}\right)^m \leq \binom{n}{k}\left(\frac{m}{n}\right)^k$ since $\frac{m}{n} \leq 1$ implies $\left(\frac{m}{n}\right)^k \geq \left(\frac{m}{n}\right)^m$, and extending the sum to $k = n$ adds non-negative terms.
+1: Recognize that $\sum_{k=0}^{n} \binom{n}{k}\left(\frac{m}{n}\right)^k = \left(1 + \frac{m}{n}\right)^n$, using the binomial theorem which states $(a+b)^n = \sum_{k=0}^{n} \binom{n}{k} a^{n-k} b^k$ with $a = 1$ and $b = \frac{m}{n}$.
+1: Apply the inequality $\left(1 + \frac{m}{n}\right)^n \leq e^m$, which follows from the fact that $(1+x)^n \leq e^{nx}$ for $x > -1$ (here $x = \frac{m}{n}$), to conclude $\frac{\sum_{k=0}^{m}\binom{n}{k}}{\left(\frac{n}{m}\right)^m} \leq e^m$, so $\sum_{k=0}^{m}\binom{n}{k} \leq \left(\frac{en}{m}\right)^m$.

Figure 5.1: Sample [Question, Answer, Rubric] tuple for evaluating LLM-as-a-judge. Correspondence between the solution and LLM-generated rubric is shown in color. Note that the rubric-generation process is able to infer logical steps not explicitly stated in the solution.

Contextual Information: Textbook Excerpt

To test the LLM's ability to reason from and apply learned concepts, rather than relying solely on its pre-trained knowledge (zero-shot learning), we include contextual information in the prompt provided to the LLM for proof generation. Specifically, for each problem, we append the chapter from the textbook immediately preceding the exercise problem. This provides the LLM with relevant definitions, theorems, and previously established results that might be instrumental in constructing the required proof, mimicking how a student might refer to recent material.

Proof Generation and Evaluation Process

The overall process is as follows: For each problem instance (Question, Staff Answer, Rubric):

1. The LLM is prompted with the **Question** and the relevant **textbook context**.
2. The LLM generates a natural language mathematical proof.
3. The generated proof is then evaluated using our **LLM-as-a-judge framework**. The framework is supplied with the generated proof, the original Question, the Staff Answer, and the detailed ground truth **Rubric** for that specific problem.
4. The LLM-as-a-judge outputs a score (and qualitative feedback) based on the rubric provided, reflecting the quality and correctness of the LLM-generated proof.

This structured approach allows us to systematically assess LLM performance across a range of difficult proof problems, with and without specific contextual grounding from the textbook. We generated the rubric using Grok 3 mini, and evaluated the proofs generated by both Grok 3 mini (xAI, 2025) and Deepseek-R1 (DeepSeek-AI et al., 2025), with Grok 3 mini serving as the judge. We used two state-of-the-art reasoning models to gauge the current frontier in mathematical reasoning. Grok 3 mini was used for the LLM-as-a-judge evaluation to maintain consistency with our work in Chapter 4.

5.2 Results

The performance of the LLM in generating proofs was evaluated on our data set. We measured the average percentage score achieved by the LLM, as determined by the LLM-as-a-judge framework using the ground truth rubrics. This was done for proofs generated with and without the appended textbook context. The results are shown in Table 5.1 below:

	average score	**fully correct %**
Grok 3 mini with context	0.571	26.0%
Grok 3 mini without context	0.443	19.5%
Deepseek-R1 with context	0.625	27.3%
Deepseek-R1 without	0.572	23.4%

Table 5.1: LLM performance on the benchmark, with and without context.

An immediate observation is that, with or without additional context, the models' performances are far from perfect. In fact, the best performing approach, Deepseek-R1 with additional context, is only able to fully complete the proofs 27.3% of the time. This suggests that the benchmark is sufficiently difficult that state-of-the-art models are unable to trivially succeed in it.

We also note that providing the proceeding chapter indeed improves the models' performance, both in terms of average score and percentage of fully correct proofs. This is in line with our expectation. This improvement warrants further research into methods to further guide LLMs to reason using a provided knowledge base, in the hopes that it can grasp the deeper mathematical concepts and relationships within the text.

Additionally, we can aggregate the performances by chapter:

Chapter number	**0**	**1**	**2**	**3**	**4**	**5**	**6**
Number of questions	2	4	23	17	15	7	9
Grok with context	0.482	1.000	0.559	0.514	0.611	0.488	0.540
Grok without context	0.482	1.000	0.365	0.430	0.449	0.447	0.396
Deepseek with context	0.875	0.719	0.603	0.603	0.607	0.659	0.633
Deepseek without context	0.794	0.928	0.494	0.471	0.599	0.727	0.589

Table 5.2: LLM performance on the benchmark, aggregated by chapters

The data suggests variability in performance based on the problem set and the availability of context. However, for most chapters, providing context led to an increase in performance for both models. These results suggest further investigation into how the nature of the problems (proof type, topic, etc.) may affect the LLM's ability to utilize context and perform complex reasoning.

Chapter 6

Discussion

This work aimed to address critical gaps in evaluating the advanced mathematical reasoning capabilities of LLMs, particularly in the domain of natural language proof generation. We introduced two primary contributions: an LLM-as-a-judge framework for assessing natural language mathematical proofs and a new benchmark of PhD-level proof-based questions evaluated using this framework. Our findings provide insights into the current strengths and, more notably, the persistent limitations of LLMs in tackling complex mathematical reasoning.

Our first major contribution, the development and evaluation of an LLM-as-a-judge framework, demonstrated considerable success. The results presented in Chapter 4 (Table 4.1 and Figure 4.2) indicate that incorporating a rubric significantly enhances the LLM's ability to evaluate mathematical proofs in alignment with human expert judgment. Critically, the framework utilizing an LLM-generated rubric also showed strong performance, achieving a Pearson correlation of 0.857 relative to the human expert evaluation. This finding is particularly significant as it addresses a key challenge highlighted in our introduction: the difficulty of grading natural language proofs consistently and at scale due to the typical reliance on manual expert evaluation. The ability to automatically generate a relevant rubric based on the problem and sample solution, and then use an LLM to judge according to this rubric, offers a promising path towards more scalable and adaptable evaluation methodologies for natural language mathematical arguments. Thus, this component provides a valuable tool for future research and benchmarking efforts in this area, moving beyond the limitations of benchmarks that focus solely on numerical outputs or require translation into formal systems.

Leveraging our LLM-as-a-judge framework (specifically, the approach using an auto-generated rubric based on the findings in Chapter 4), the second component of our study assessed the capabilities of current LLMs in generating natural language proofs for our new benchmark. The results from this investigation, detailed in Chapter 5 (Table 5.1), suggest further room for improvement. Despite the remarkable advancements in LLM architecture and prompting techniques, the performance on these complex proof-based tasks indicates that current models, such as the Grok 3 mini and Deepseek-R1 used in our experiments, are generally still unable to adequately complete them. The average percentage scores achieved, even with contextual information, often remained low across various chapters of the advanced textbook from which problems were sourced. Many chapters show average scores around 40-60%. On the other hand, this is evidence that our benchmark is challenging enough to be valuable in evaluating models' mathematical reasoning capabilities.

These findings lend support to the concerns raised in the introduction regarding a potential "reasoning illusion," where success on benchmarks emphasizing numerical answers or less complex problem structures might not fully translate to an ability to construct intricate arguments that require deep mathematical insight. This emphasizes the continued necessity for challenging benchmarks, like the one introduced, that probe these deeper reasoning abilities and provide a more accurate measure of LLM capabilities.

6.1 Limitations

While our study provides valuable insights into both LLM-based evaluation and generation of mathematical proofs, it is important to acknowledge its limitations. Firstly, our LLM-as-a-judge evaluation done in Chapter 4 was conducted using only one model, Grok 3 mini. While a capable model, its performance might not be representative of all LLMs, which themselves are rapidly evolving. Different architectures or models trained with different data or methodologies might yield different results on our benchmark.

Our LLM-as-a-judge framework, while showing promising correlation with human scores, is not a perfect substitute for nuanced human expert evaluation. Mathematical reasoning can be subtle, and LLM judges may still possess inherent biases. The models also may not fully comprehend more nuanced logical flaws or may discount novel correct approaches not covered by the sample solution. The high Pearson correlations, while indicative of a strong positive relationship, is only the first step towards an LLM-as-a-judge proof evaluation format that is suitable for wide-scale use.

Additionally, we evaluated our LLM-as-a-judge framework using undergraduate linear algebra proofs. The LLM-generated rubric approach we settled on may not perform to the same capacity for more complex mathematical arguments such as those included in our benchmark.

Our new benchmark, while comprising 77 challenging PhD-level questions, represents a specific selection of problems from a particular domain (high-dimensional probability and related linear algebra). The performance observed might vary with problems from different mathematical fields or of different styles.

Chapter 7

Conclusion and Future Directions

In conclusion, this paper has presented an LLM-as-a-judge framework that demonstrates significant promise for the scalable evaluation of natural language mathematical proofs. This framework shows high agreement with human expert graders, particularly when guided by rubrics, including those automatically generated by an LLM. This addresses a critical need for efficient evaluation methods in mathematical AI. However, when this framework was applied to assess LLM-generated proofs for our new, challenging benchmark of 77 PhD-level problems, we find that current leading LLMs, despite recent advancements, are largely still unable to fully and reliably solve these complex tasks. The significant improvement derived from naively providing contextual information suggests that the models are, at some level, leveraging the text and building mathematical knowledge.

These findings have significant implications. They reaffirm the necessity for rigorous benchmarks that test the limits of AI's mathematical abilities. These benchmarks should move beyond surface-level assessments and instead fundamentally gauge the model's mathematical understanding. Our development of an LLM-as-a-judge framework for natural language mathematical arguments offers a practical tool for researchers by providing a more efficient means of evaluation than manual grading. This may accelerate the development, testing, and iteration of new models and techniques.

Looking ahead, these insights pave the way for several crucial directions for future research:

- **Refining LLM-as-a-Judge Systems:** While our rubric-based approach is promising, efforts should continue towards achieving even higher alignment with human expert evaluation. This may involve additional fine-tuning or RL methods. Additionally, a particularly important and challenging extension would be to develop and assess the efficacy of LLM-as-a-judge frameworks for mathematical proofs in scenarios where a ground truth sample solution does not exist. This capability would be invaluable for evaluating novel conjectures or student solutions to unseen problems. However, this would require the LLM-judge to evaluate correctness and rigor based on fundamental mathematical principles and the problem statement alone, which would be a daunting task.

- **Expanding Benchmark Scope and Robustness:** The continuous development and expansion of diverse and challenging proof-based benchmarks are essential. While our benchmark of 77 PhD-level questions provides a valuable testbed, future work must involve more robust testing across a wider range of mathematical textbooks, domains (e.g., abstract algebra, topology, analysis), and difficulty levels. This will help ensure that developed capabilities are generalizable and not specific to the style or content of a single source or narrow field. This will help foster more universally capable mathematical AI and mitigate issues such as benchmark saturation.

- **Enhancing Intrinsic LLM Reasoning Capabilities:** There is a clear need to improve the fundamental mathematical reasoning skills of LLMs. This could involve exploring novel architectures, developing training paradigms that explicitly focus on logical deduction and proof structure, or advancing neuro-symbolic approaches. Future work should investigate more sophisticated modeling methods to improve LLM ability on these proofs by more efficiently leveraging the provided knowledge base (in our case, the textbook), perhaps by training models to iteratively understand and build upon concepts, mimicking a structured human learning process.

Bibliography

Ahn, J., Verma, R., Lou, R., Liu, D., Zhang, R., & Yin, W. (2024). Large language models for mathematical reasoning: Progresses and challenges. https://arxiv.org/abs/2402.00157

Amini, A., Gabriel, S., Lin, P., Koncel-Kedziorski, R., Choi, Y., & Hajishirzi, H. (2019). Mathqa: Towards interpretable math word problem solving with operation-based formalisms. https://arxiv.org/abs/1905.13319

Budagam, D., Kumar, A., Khoshnoodi, M., KJ, S., Jain, V., & Chadha, A. (2024). Hierarchical prompting taxonomy: A universal evaluation framework for large language models aligned with human cognitive principles. https://arxiv.org/abs/2406.12644

Cheng, J., & Durme, B. V. (2024). Compressed chain of thought: Efficient reasoning through dense representations. https://arxiv.org/abs/2412.13171

Chervonyi, Y., Trinh, T. H., Olšák, M., Yang, X., Nguyen, H., Menegali, M., Jung, J., Verma, V., Le, Q. V., & Luong, T. (2025). Gold-medalist performance in solving olympiad geometry with alphageometry2. https://arxiv.org/abs/2502.03544

Cobbe, K., Kosaraju, V., Bavarian, M., Chen, M., Jun, H., Kaiser, L., Plappert, M., Tworek, J., Hilton, J., Nakano, R., Hesse, C., & Schulman, J. (2021). Training verifiers to solve math word problems. https://arxiv.org/abs/2110.14168

Davoodi, A. G., Davoudi, S. P. M., & Pezeshkpour, P. (2025). Llms are not intelligent thinkers: Introducing mathematical topic tree benchmark for comprehensive evaluation of llms. https://arxiv.org/abs/2406.05194

DeepSeek-AI, Guo, D., Yang, D., Zhang, H., Song, J., Zhang, R., Xu, R., Zhu, Q., Ma, S., Wang, P., Bi, X., Zhang, X., Yu, X., Wu, Y., Wu, Z. F., Gou, Z., Shao, Z., Li, Z., Gao, Z., . . . Zhang, Z. (2025). Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. https://arxiv.org/abs/2501.12948

Dong, K., & Ma, T. (2025). Stp: Self-play llm theorem provers with iterative conjecturing and proving. https://arxiv.org/abs/2502.00212

Forootani, A. (2025). A survey on mathematical reasoning and optimization with large language models. https://arxiv.org/abs/2503.17726

Frieder, S., Pinchetti, L., Chevalier, A., Griffiths, R.-R., Salvatori, T., Lukasiewicz, T., Petersen, P. C., & Berner, J. (2023). Mathematical capabilities of chatgpt. https://arxiv.org/abs/2301.13867

Gao, B., Cai, Z., Xu, R., Wang, P., Zheng, C., Lin, R., Lu, K., Liu, D., Zhou, C., Xiao, W., Hu, J., Liu, T., & Chang, B. (2024). Llm critics help catch bugs in mathematics:

Towards a better mathematical verifier with natural language feedback. https://arxiv.org/abs/2406.14024

Gao, B., Song, F., Yang, Z., Cai, Z., Miao, Y., Dong, Q., Li, L., Ma, C., Chen, L., Xu, R., Tang, Z., Wang, B., Zan, D., Quan, S., Zhang, G., Sha, L., Zhang, Y., Ren, X., Liu, T., & Chang, B. (2025). Omni-MATH: A universal olympiad level mathematic benchmark for large language models. *The Thirteenth International Conference on Learning Representations*. https://openreview.net/forum?id=yaqPf0KAlN

Glazer, E., Erdil, E., Besiroglu, T., Chicharro, D., Chen, E., Gunning, A., Olsson, C. F., Denain, J.-S., Ho, A., de Oliveira Santos, E., Järviniemi, O., Barnett, M., Sandler, R., Vrzala, M., Sevilla, J., Ren, Q., Pratt, E., Levine, L., Barkley, G., . . . Wildon, M. (2024). Frontiermath: A benchmark for evaluating advanced mathematical reasoning in ai. https://arxiv.org/abs/2411.04872

Gu, J., Jiang, X., Shi, Z., Tan, H., Zhai, X., Xu, C., Li, W., Shen, Y., Ma, S., Liu, H., Wang, S., Zhang, K., Wang, Y., Gao, W., Ni, L., & Guo, J. (2025). A survey on llm-as-a-judge. https://arxiv.org/abs/2411.15594

Gulati, A., Ladsaria, D., Mishra, S., Sidhu, J., & Miranda, B. (2024). An evaluation benchmark for autoformalization in lean4. https://arxiv.org/abs/2406.06555

Hao, S., Sukhbaatar, S., Su, D., Li, X., Hu, Z., Weston, J., & Tian, Y. (2024). Training large language models to reason in a continuous latent space. https://arxiv.org/abs/2412.06769

Hendrycks, D., Burns, C., Kadavath, S., Arora, A., Basart, S., Tang, E., Song, D., & Steinhardt, J. (2021). Measuring mathematical problem solving with the math dataset. https://arxiv.org/abs/2103.03874

Hong, Z., Wu, H., Dong, S., Dong, J., Xiao, Y., Zhang, Y., Wang, Z., Huang, F., Li, L., Yang, H., & Huang, X. (2025). Benchmarking large language models via random variables. https://arxiv.org/abs/2501.11790

Huang, K., Guo, J., Li, Z., Ji, X., Ge, J., Li, W., Guo, Y., Cai, T., Yuan, H., Wang, R., Wu, Y., Yin, M., Tang, S., Huang, Y., Jin, C., Chen, X., Zhang, C., & Wang, M. (2025). Mathperturb: Benchmarking llms' math reasoning abilities against hard perturbations. https://arxiv.org/abs/2502.06453

Huang, W., Jia, B., Zhai, Z., Cao, S., Ye, Z., Zhao, F., Xu, Z., Hu, Y., & Lin, S. (2025). Vision-r1: Incentivizing reasoning capability in multimodal large language models. https://arxiv.org/abs/2503.06749

Krumdick, M., Lovering, C., Reddy, V., Ebner, S., & Tanner, C. (2025). No free labels: Limitations of llm-as-a-judge without human grounding. https://arxiv.org/abs/2503.05061

Li, Z., Xu, X., Shen, T., Xu, C., Gu, J.-C., Lai, Y., Tao, C., & Ma, S. (2024). Leveraging large language models for nlg evaluation: Advances and challenges. https://arxiv.org/abs/2401.07103

Lin, Y., Tang, S., Lyu, B., Wu, J., Lin, H., Yang, K., Li, J., Xia, M., Chen, D., Arora, S., & Jin, C. (2025). Goedel-prover: A frontier model for open-source automated theorem proving. https://arxiv.org/abs/2502.07640

Ling, Z., Fang, Y., Li, X., Huang, Z., Lee, M., Memisevic, R., & Su, H. (2023). Deductive verification of chain-of-thought reasoning. https://arxiv.org/abs/2306.03872

Liu, W., Hu, H., Zhou, J., Ding, Y., Li, J., Zeng, J., He, M., Chen, Q., Jiang, B., Zhou, A., & He, L. (2025). Mathematical language models: A survey. https://arxiv.org/abs/2312.07622

Miao, N., Teh, Y. W., & Rainforth, T. (2023). Selfcheck: Using llms to zero-shot check their own step-by-step reasoning. https://arxiv.org/abs/2308.00436

Moshkov, I., Hanley, D., Sorokin, I., Toshniwal, S., Henkel, C., Schifferer, B., Du, W., & Gitman, I. (2025). Aimo-2 winning solution: Building state-of-the-art mathematical reasoning models with openmathreasoning dataset. https://arxiv.org/abs/2504.16891

Petrov, I., Dekoninck, J., Baltadzhiev, L., Drencheva, M., Minchev, K., Balunović, M., Jovanović, N., & Vechev, M. (2025). Proof or bluff? evaluating llms on 2025 usa math olympiad. https://arxiv.org/abs/2503.21934

Romera-Paredes, B., Barekatain, M., Novikov, A., Balog, M., Kumar, M. P., Dupont, E., Ruiz, F. J. R., Ellenberg, J. S., Wang, P., Fawzi, O., Kohli, P., & Fawzi, A. (2024). Mathematical discoveries from program search with large language models. *Nature*, *625*(7995), 468–475. https://doi.org/10.1038/s41586-023-06924-6

Stephan, A., Zhu, D., Aßenmacher, M., Shen, X., & Roth, B. (2025). From calculation to adjudication: Examining llm judges on mathematical reasoning tasks. https://arxiv.org/abs/2409.04168

Tian, X., Zhao, S., Wang, H., Chen, S., Ji, Y., Peng, Y., Zhao, H., & Li, X. (2025). Think twice: Enhancing llm reasoning by scaling multi-round test-time thinking. https://arxiv.org/abs/2503.19855

Trinh, T. H., Wu, Y., Le, Q. V., He, H., & Luong, T. (2024). Solving olympiad geometry without human demonstrations. *Nature*, *625*(7995), 476–482. https://doi.org/10.1038/s41586-023-06747-5

Tsoukalas, G., Lee, J., Jennings, J., Xin, J., Ding, M., Jennings, M., Thakur, A., & Chaudhuri, S. (2024). Putnambench: Evaluating neural theorem-provers on the putnam mathematical competition. https://arxiv.org/abs/2407.11214

Vershynin, R. (2018). *High-dimensional probability: An introduction with applications in data science*. Cambridge University Press.

Wang, H., Unsal, M., Lin, X., Baksys, M., Liu, J., Santos, M. D., Sung, F., Vinyes, M., Ying, Z., Zhu, Z., Lu, J., de Saxcé, H., Bailey, B., Song, C., Xiao, C., Zhang, D., Zhang, E., Pu, F., Zhu, H., … Li, J. (2025). Kimina-prover preview: Towards large formal reasoning models with reinforcement learning. https://arxiv.org/abs/2504.11354

Wang, Q., Lou, Z., Tang, Z., Chen, N., Zhao, X., Zhang, W., Song, D., & He, B. (2025). Assessing judging bias in large reasoning models: An empirical study. https://arxiv.org/abs/2504.09946

Wataoka, K., Takahashi, T., & Ri, R. (2024). Self-preference bias in llm-as-a-judge. https://arxiv.org/abs/2410.21819

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E., Le, Q., & Zhou, D. (2023). Chain-of-thought prompting elicits reasoning in large language models. https://arxiv.org/abs/2201.11903

xAI. (2025). Introducing the Grok 3 family: Advancing AI with unparalleled reasoning and efficiency [Accessed: 2025-05-15].

Xu, X., Xu, Y., Chen, T., Yan, Y., Liu, C., Chen, Z., Wang, Y., Yin, Y., Wang, Y., Shang, L., & Liu, Q. (2025). Teaching llms according to their aptitude: Adaptive reasoning for mathematical problem solving. https://arxiv.org/abs/2502.12022

Xu, Y., Guo, X., Zeng, Z., & Miao, C. (2025). Softcot: Soft chain-of-thought for efficient reasoning with llms. https://arxiv.org/abs/2502.12134

Yang, K., Poesia, G., He, J., Li, W., Lauter, K., Chaudhuri, S., & Song, D. (2024). Formal mathematical reasoning: A new frontier in ai. https://arxiv.org/abs/2412.16075

Yu, Z., Peng, R., Ding, K., Li, Y., Peng, Z., Liu, M., Zhang, Y., Yuan, Z., Xin, H., Huang, W., Wen, Y., Zhang, G., & Liu, W. (2025). Formalmath: Benchmarking formal mathematical reasoning of large language models. https://arxiv.org/abs/2505.02735

Zhang, H., Da, J., Lee, D., Robinson, V., Wu, C., Song, W., Zhao, T., Raja, P. V., Zhuang, C., Slack, D. Z., Lyu, Q., Hendryx, S. M., Kaplan, R., Lunati, M., & Yue, S. (2024). A careful examination of large language model performance on grade school arithmetic. *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*. https://openreview.net/forum?id=RJZRhMzZzH

Zhang, J., Wang, Q., Ji, X., Liu, Y., Yue, Y., Zhang, F., Zhang, D., Zhou, G., & Gai, K. (2025). Leanabell-prover: Posttraining scaling in formal reasoning. https://arxiv.org/abs/2504.06122

Zhao, C., Silva, M., & Poulsen, S. (2025). Autograding mathematical induction proofs with natural language processing. https://arxiv.org/abs/2406.10268

Zhao, X., Zheng, L., Bo, H., Hu, C., Thakker, U., & Kong, L. (2024). Subgoalxl: Subgoal-based expert learning for theorem proving. https://arxiv.org/abs/2408.11172

Zheng, K., Han, J. M., & Polu, S. (2022). Minif2f: A cross-system benchmark for formal olympiad-level mathematics. https://arxiv.org/abs/2109.00110

Zheng, L., Chiang, W.-L., Sheng, Y., Zhuang, S., Wu, Z., Zhuang, Y., Lin, Z., Li, Z., Li, D., Xing, E. P., Zhang, H., Gonzalez, J. E., & Stoica, I. (2023). Judging llm-as-a-judge with mt-bench and chatbot arena. https://arxiv.org/abs/2306.05685