An in-depth human study of the mathematical reasoning abilities in Large Language Models

Carolina Dias-Alexiou, Edison Marrese-Taylor, Yutaka Matsuo
Graduate School of Engineering, The University of Tokyo
{carolina.dias,emarrese,matsuo}@weblab.t.u-tokyo.ac.jp

Abstract

We study the generalization capabilities of large language models (LLM) through the lens of mathematical reasoning, asking if these models can recognize that two structures are the same even when they do not share the same nomenclature. We propose a human study to evaluate if LLMs reproduce proofs that they have most likely seen during training, but when the symbols do not match the ones seen. To test this in a controlled scenario, we look at proofs in *propositional calculus*, foundational for other logic systems, semantically complete and widely discussed online. We replace the implication operator ($\rightarrow$) with an unrelated, arbitrary symbol (♠) and ask experts to evaluate how the output of a selection of LLMs changes in terms of compliance, correctness, extensiveness and coherence. Our results show that nearly all our tested models produce lower quality proofs in this test, in particular open-weights models, suggesting the abilities of these LLMs to reason in this context have important limitations.

1 Introduction

Mathematical reasoning is a key aspect of human intelligence that encompasses pattern recognition and logical entailment. The development of artificial intelligence systems capable of tasks such as solving applied and theoretical mathematical problems has been a long-standing focus of research in the fields of machine learning and natural language processing, dating back to the 1960s (Feigenbaum and Feldman, 1963; Bobrow, 1964).

Our interest in this topic rises from recent attempts to use language models for theorem proving by means of ITPs' programming languages and databases of theorems with their proofs. Deep learning models can be trained in one of these many programming languages, and then used to generate mathematical proofs. Data sources for neural theorem proving in ITPs include interactive learning environments that interface with ITPs, and datasets derived from proofs in ITP libraries.

When it comes to mathematical reasoning, and in particular to the ability of models to understand logical statements, we note that despite the abundance of studies, previous works generally assume that only "variables" are the ones not constant throughout problems. However, we see that in mathematics different nomenclatures are used in different areas, as well as in different time periods, to express the same ideas. Examples of this fact include the use of $\supset$ and $\rightarrow$ to denote implication; Leibniz's $\frac{\mathrm{d}f}{\mathrm{d}x}$, Lagrange's f', and Newton's $\dot{f}$ notation in differential calculus; the different notation for the inner product for physicists $\langle\cdot|\cdot\rangle$ and for mathematicians $\langle\cdot,\cdot\rangle$; prefix $f(x)$ and postfix $(x)f$ notations for functions; and different notations for the von Neumann generated algebra, such as: $W^*(\cdot,\cdot)$ and $\cdot \vee \cdot$, to name a few.

As we attempt to use LLMs to tackle proof generation tasks, for example using the "informal theorem proving" approach (please see §2 for more details on this), we think researchers and practitioners need to take this fact into consideration. Furthermore, and in contrast to all these models, current state-of-the-art models in NLP are trained on large datasets of text extracted from the web. In this case, we have limited or no control on the kinds of expressions and/or operators that the models are exposed to during training. We can, however, still assume the models have been exposed mostly to the standard nomenclature.

Given this scenario, we ask: *can these models recognize that two structures are the same even if they do not share the same nomenclature?* For example, can models reproduce proofs that they have most likely seen during training, but when the symbols do not match the ones seen? If so, to what extent are these proofs plausible and correct? In order to answer these questions, we perform

Proceedings of The 3rd Workshop on Mathematical Natural Language Processing (MathNLP 2025), pages 186–194

an in-depth human evaluation to assess the quality of the output generated by LLMs when prompted to generate proofs similar to the ones seen during training but with expressions that use a different notation.

2 Related Work

Automated Theorem Proving The first proofs using computers started appearing as early as the 1950s, soon after electronic computers became available. This played a big role in the development of the field of automated reasoning, which itself led to the development of AI. Most of the early work on computer-assisted proof was devoted to *automated theorem proving* (ATP) (Harrison et al., 2014), in which machines were expected to prove assertions fully automatically. The increased availability of interactive time-sharing computer operating systems in the 1960s allowed the development of *interactive theorem provers* (ITPs) in which the machine and the user work together to produce a formal proof. While ATPs include proof-search algorithms to generate whole proofs, ITPs usually check the validity of human input statements, although they may also include reduced automated tools.

More recently, work on "informal" theorem proving has presented an alternative medium for theorem proving, in which statements and proofs are written in a mixture of natural language and symbols used in "standard" mathematics (e.g., in LaTeX), and are checked for correctness by humans. Here we find the work of Welleck et al. (2021) who developed NaturalProofs, a large-scale dataset of 32k informal mathematical theorems, definitions, and proofs, and provided a benchmark for premise selection via retrieval and generation tasks. Most of the data is taken from websites like `proofwiki.com`, and though this enables more flexibility when proving, the task is approached in a way similar to ITPs.

Mathematical Reasoning in LLMs Early works attempting to study the ability of models to recognize patterns in mathematical expressions focused on the manipulation of simple expressions using standard notation. For example, Allamanis et al. (2017) trained models on datasets in which pairs of examples contain Boolean logic and arithmetic expressions which are known to be equivalent. For example, expressions like c^2 and $(c \cdot c) + (b - b)$ are equivalent. However, expressions with the same structure, but different variables, such as $c \cdot (a \cdot a + b)$ and $f \cdot (d \cdot d + e)$, are not. They showed that such models were capable of relating non-paired expressions, like $a - (b - c)$ and $b - (a + c)$, as negations of each other.

Evans et al. (2018) studied the ability of neural networks to understand logical entailment via training models on synthetic datasets of logical statements and their evaluations (True/False). Concretely, they generated datasets of triples of the form $(A, B, A \vDash B)$, where A and B are formulas of propositional logic, and $A \vDash B$ is 1 if A entails B, and 0 otherwise. They concluded that, from the models available at the time, those with a tree structure seemed to be better for domains with unambiguous syntax.

In these works, expressions are generated automatically starting from a set of simple rules plus a set of arbitrary combinations. This allows to scale and control the types of expressions that are shown to the model during training/inference. Later Cobbe et al. (2021) and Rein et al. (2023) shifted to a question-answer format using natural language, with the release of the GSM8K and GPQA datasets, respectively, while also extending the class of questions to other areas of mathematics, like calculus and probability, where the ability to control for the type of expressions is reduced.

3 Proposed Approach

To study the posed research questions under a controlled scenario, we look at proofs in *propositional calculus*, a branch of formal logic that deals with propositions, which can be true or false, and relations between propositions, including the construction of arguments based on them (Wrenn, 2025). Propositional calculus, also known as zeroth-order logic, does not deal with quantifiers over non-logical objects (unlike first-order or higher-order logic). There are several reasons that we think make this the ideal scenario for our study: (1) All the machinery of propositional logic is included in first-order logic, higher-order logic, and all mathematics. In this sense, propositional logic is the foundation of other logic systems; (2) Propositional calculus is semantically complete, i.e. any tautology (true formulas) can be proved with the formal axioms and the rules of inference of the system; (3) Being the subject of common undergraduate courses, demonstrations in this context have been widely discussed online (for example, in fora such

as Math Stack Exchange) so we can reasonably assume that LLMs have been exposed to these types of proofs, and (4) It is a minimalist setup which allows us to include the entire logical structure in the prompt, thus reducing the amount of assumptions needed.

Propositional calculus is typically studied with a formal system, which contains a formal language and a deductive system. The language is composed of a set of well-formed formulas, which are strings of symbols from an alphabet (composed of propositional variables and propositional connectives) formed by a formal grammar (formation rules). The deductive system, in turn, contains the rules of inference, a function which takes premises and returns conclusions. To assess how models generalize in this scenario, we compared proofs generated by these models using "usual" and "unusual" symbols for connectives.

We use a standard proof system usually referred to as a Hilbert system. This is a deductive system that generates theorems from axioms (tautologies taken as starting point for further reasoning) and *modus ponens*. *Modus ponens* can be summarized as: If P implies Q and P is known to be true, one can conclude that Q must also be true. It is generally expressed as $\{P \rightarrow Q,\ P\} \vdash Q$, where the turnstile symbol ($\vdash$) denotes derivability, i.e. there is a formal derivation of a theorem from the axioms. As for connectives, we limit it to the logical and ($\wedge$), logical or ($\vee$), the negation operator ($\neg$), and the implication operator ($\rightarrow$). For the axioms, we use a common set of 14 axioms used in undergraduate courses, shown in Figure 1.

For our study, we propose to replace the implication operator ($\rightarrow$) with an unrelated, arbitrary symbol (♠). In order to produce a significant perturbation in the input token distribution, we specifically select the unicode representation of the symbol (U+2660) for the replacement. Alternative replacements are left for future work. We select two common theorems from propositional calculus extracted from Rossegger (2019), as shown in Figure 2 and test models in two different scenarios, as follows.

Full Context (FC) Our first evaluation scheme is intended to simulate a noisy retrieval step prior to the proof generation. Concretely, we offer the model the complete set of axioms together with the selected rule of inference, *modus ponens*. Thus, in this scenario, we can also test the model's ability to identify and retrieve only the axioms that are needed to prove the selected theorem.

Axioms

(Ax 1) : $((\phi \wedge \psi) \rightarrow \phi)$
(Ax 2) : $((\phi \wedge \psi) \rightarrow \psi)$
(Ax 3) : $(\phi \rightarrow (\psi \rightarrow (\phi \wedge \psi)))$
(Ax 4) : $(\phi \rightarrow (\phi \vee \psi))$
(Ax 5) : $(\phi \rightarrow (\psi \vee \phi))$
(Ax 6) : $((\phi \rightarrow \chi) \rightarrow ((\psi \rightarrow \chi) \rightarrow ((\phi \vee \psi) \rightarrow \chi)))$
(Ax 7) : $(\phi \rightarrow (\psi \rightarrow \phi))$
(Ax 8) : $((\phi \rightarrow (\psi \rightarrow \chi)) \rightarrow ((\phi \rightarrow \psi) \rightarrow (\phi \rightarrow \chi)))$
(Ax 9) : $((\phi \rightarrow \psi) \rightarrow ((\phi \rightarrow \neg\psi) \rightarrow \neg\phi))$
(Ax 10) : $(\neg\phi \rightarrow (\phi \rightarrow \psi))$
(Ax 11) : $(\phi \vee \neg\phi)$
(Ax 12) : $((\phi \wedge \neg\phi) \rightarrow \psi)$
(Ax 13) : $((\phi \rightarrow (\psi \wedge \neg\psi)) \rightarrow \neg\phi)$
(Ax 14) : $(\neg\neg\phi \rightarrow \phi)$
(MP) : $\{P \rightarrow Q, P\} \vdash Q$

Figure 1: Portion of the prompt provided to the LLMs showing the content of the full context provided, namely, the axioms and rules of inference we allow the models to use.

Selected Context (SC) We assume that the relevant axioms for the requested proof have already been selected by an oracle, and we offer only these axioms and rule of inference to the model input. For each question, we manually select the axioms required (Axioms 6, 7, 10 for Question A; Axioms 7, 8 for Question B). In practice, we reassign identifier numbers to them, always starting from 1, to avoid ambiguity.

A key point of our study is to ensure that the generated proofs are checked by mathematicians. Previous work has stressed the need to rely on experts for evaluation of theorem proving systems, including the work of Welleck et al. (2022), who carried on an in-depth annotation where an expert annotator is presented with the theorem, proof-so-far, and a generated next-step. Frieder et al. (2023) also highlight that human evaluation of advanced mathematics that approaches research level is expensive and requires experts. The evaluation of the output of the language model for their work was performed by the authors, who are all mathematicians.

To perform the evaluation, we concretely rely on a volunteer (one of the authors of this paper) who has a Master's degree in mathematics. We design an annotation interface where for each case, we show the annotator the exact input fed to the model,

Question A

Question: Prove that

$$\vdash ((\neg P \vee Q) \rightarrow (P \rightarrow Q))$$

Answer:

$$\begin{gathered}((\neg P \rightarrow (P \rightarrow Q)) \rightarrow ((Q \rightarrow (P \rightarrow Q)) \rightarrow \\ ((\neg P \vee Q) \rightarrow (P \rightarrow Q)))) \quad \text{(Ax 6)} \\ (\neg P \rightarrow (P \rightarrow Q)) \quad \text{(Ax 10)} \\ ((Q \rightarrow (P \rightarrow Q)) \rightarrow ((\neg P \vee Q) \rightarrow (P \rightarrow Q))) \quad \text{1, 2 MP} \\ (Q \rightarrow (P \rightarrow Q)) \quad \text{(Ax 7)} \\ ((\neg P \vee Q) \rightarrow (P \rightarrow Q)) \quad \text{3, 4 MP}\end{gathered}$$

Question B

Question: Prove that

$$\{(P \rightarrow Q), (Q \rightarrow R)\} \vdash (P \rightarrow R)$$

Answer:

$$\begin{gathered}(P \rightarrow Q) \quad \text{(hyp)} \\ (Q \rightarrow R) \quad \text{(hyp)} \\ ((Q \rightarrow R) \rightarrow (P \rightarrow (Q \rightarrow R))) \quad \text{(Ax 7)} \\ (P \rightarrow (Q \rightarrow R)) \quad \text{2, 3 MP} \\ ((P \rightarrow (Q \rightarrow R)) \rightarrow ((P \rightarrow Q) \rightarrow (P \rightarrow R))) \quad \text{(Ax 8)} \\ ((P \rightarrow Q) \rightarrow (P \rightarrow R)) \quad \text{4, 5 MP} \\ (P \rightarrow R) \quad \text{1, 6 MP}\end{gathered}$$

Figure 2: Details of the question (top: Question A, bottom: Question B) utilized for our study, also showing a possible proof which we allow our annotators to see.

as well as the output generated by it. Below, we list the tasks that we require our annotator to perform.

- First, we ask the annotators if the output of the model contains a proof.
- We present the steps of a correct proof and ask the annotators to judge if each step appears in the output of the model. Additionally, if the step invokes an axiom or rule of inference, we ask them to do the following.
  - If the step invokes an axiom, we ask to check if variables were substituted correctly.
  - If the step uses a rule of inference, we ask to check if the rule is properly invoked.
  - We ask the annotator to judge whether the step contributes to the proof in the sense that it stirs the overall flow of the proof in the right direction towards conclusion.
- With respect to the clarity of the overall text, we ask the annotators to rate the output on a scale of 0 to 4 via the following labels: "Very Incoherent, Incoherent, Neither Coherent nor Incoherent, Coherent, Very Coherent".
- In order to evaluate the compliance to the task, we ask whether the proof attempts to use information other than the necessary elements that were provided to the model as input. Concretely, we ask the annotators to indicate if the proof uses an additional axiom, if this axiom was provided in the input, and if it uses an additional rule of inference, or an additional hypothesis.
- Finally, we also allow the annotators to freely provide us specific feedback by highlighting spans of the model output that calls their attention, to which they can add free-text comments.

The above questions were designed to incorporate most, if not all, aspects that one would take into consideration when grading the same questions in an exam.

4 Results

For this study, we consider the following models: (1) API-based LLMs, including ChatGPT (*gpt-3.5-turbo-0125*) (Brockman et al., 2023), Claude 3 Opus (*claude-3-opus-20240229*) (Anthropic, 2023), (2) Open-weights models, including Llama 3 (*Meta-Llama-3-8B-Instruct*) (Grattafiori and the Llama 3 Team, 2024; AI@Meta, 2024), Llama 3.1 (*Llama-3.1-8B-Instruct*) (Int) and Gemma 2 (*gemma-2-9b-it*) (Team, 2024). The latter models are obtained from HuggingFace, and quantized to 4-bits (Dettmers et al., 2023) to fit our GPU memory. For each input, we obtain 3 outputs from each model using a different random seed. We computed the following metrics to summarize model behavior and measure performance.

- Percentage of times the model generated output that contains a proof and where the model did not utilize a hypothesis, axiom, or rule of inference other than those provided. We consider this a measure of the model compliance with our instruction (Compliance).
- Percentage of steps from the gold standard proof that appear in the generated proof, i.e., to what extent the model used the needed axioms and *modus ponens* (Extensiveness).

Model	Ctx.	Compliance		Extensiveness		Correctness		Flow		Clarity	
		→	♠	→	♠	→	♠	→	♠	→	♠
GPT-3.5-turbo	SC	66.67%	**83.33%**	**62.38%**	49.52%	**30.00%**	6.67%	**33.33%**	20.00%	**2.67**	2.33
	FC	50.00%	**100%**	**36.19%**	30.48%	**10.00%**	6.67%	**26.67%**	16.67%	**3.00**	2.60
Claude 3 Opus	SC	**100%**	**100%**	**95.24%**	93.14%	**86.67%**	72.00%	**93.33%**	84.00%	**3.83**	3.40
	FC	**100%**	83.33%	**85.71%**	75.71%	**66.67%**	33.33%	**80.00%**	50.00%	**3.67**	3.00
Gemma 2 (9B)	SC	0.00%	0.00%	**11.90%**	0.00%	0.00%	0.00%	3.33%	0.00%	**4.00**	-
	FC	0.00%	0.00%	**16.67%**	0.00%	0.00%	0.00%	3.33%	0.00%	**4.00**	-
Llama 3 (8B)	SC	50.00%	**66.67%**	**49.52%**	41.43%	**3.33%**	**3.33%**	**16.67%**	6.67%	**1.83**	1.80
	FC	33.33%	**83.33%**	**35.71%**	21.90%	**10.00%**	3.33%	**13.33%**	0.00%	**2.17**	1.60
Llama 3.1 (8B)	SC	33.33%	**100%**	**42.38%**	40.95%	0.00%	**3.33%**	**6.67%**	**6.67%**	**1.17**	**1.17**
	FC	66.67%	**50.00%**	**32.38%**	24.29%	**3.33%**	0.00%	**6.67%**	**6.67%**	**2.00**	1.50

Table 1: Summary of the results of our human evaluation study, where Ctx. is short for context. Bold numbers indicate the best score for each pair of (→, ♠) prompts for a given model, and we underline the best score across the FC and SC scenarios for each model.

- Percentage of steps that appear in the generated proof and were correctly applied. In other words, the steps that are correct (Correctness).
- Percentage of steps that appear in the output providing an expression which is a step towards finalizing the proof, even if it was not correctly deduced (Flow).
- Average clarity score reported for the overall text output from a given model (Clarity).

Table 1 summarizes the results of our evaluation. With the exception of Compliance, all metrics show a similar trend of decrease in model performance when the questions are perturbed with ♠. The same is also true when we compare the Full Context to the Selected Context. Most interestingly, the score for Flow was always higher than the Correctness, indicating that models tend to recall or retrieve parts of the right answer from memory, even if they cannot follow the correct sequence of logical steps.

We also observe that not only Compliance did not follow the decrease trend seen for other metrics, but also that most models showed an increase in this metric when prompted with the arbitrary symbol. In fact, this variation in Compliance appears to strongly correlate with our ♠-based replacement scenario. Overall, the average compliance with the original symbol (→) is 43.85%, which compares unfavorably against 71.89% for our replacement scenario (♠). We note that such a significant gap is not evident when looking at performance differences due to variations in context, where we observe an average Compliance of 60.22% for SC and 63.39% for FC. Since invoking facts outside of the givens in the prompt is a big factor in how we compute the Compliance criteria, we believe this counterintuitive increase in value can be at least partially explained by the changes in the semantics induced by our symbol replacement. Intuitively, the semantic similarity between these expressions and the examples seen during training should be lower, making it more unlikely for the model to retrieve relevant content seen during training, even if not useful for the proof.

The model with the best performance across all measures was Claude 3 Opus. Like for most models, its Extensiveness measure was subject to a decrease when one compares the easiest scenario (SC with →) to the hardest scenario (FC with ♠), but its lowest value (75.71%) was still higher than the highest Extensiveness measure of any other model. However, it did experience a sharp decrease in both Correctness (from 86.67% to 33.33%) and Flow (from 93.33% to 50.00%), with the first being the largest difference in performance of any metric for any model. As table 1 shows, there is also a big difference in performance between the API-based LLMs and the Open-weights models. The Gemma 2 model refused to provide a proof in most cases, as it can be seeing by its Compliance measure, making it impossible to draw conclusions about its abilities. We think these results suggests open-weights models are significantly behind APIs, which is well-aligned with performance measured in popular automatic benchmarks.

To delve further into the data, we split the Clarity into intervals, as seen in Table 3, and looked at the other metrics' behavior in each range (here we excluded the cases in which the model did not pro-

Model	Metric	Scenario Δ	
		→ to ♠	SC to FC
GPT-3.5-turbo	Compliance	33.33%	0.00%
	Extensiveness	-9.29%	-22.62%
	Correctness	-13.33%	-10.00%
	Flow	-11.67%	-5.00%
	ExtraHyp	-25.00%	8.33%
	ExtraAxiom	0.00%	83.33%
	ExtraROI	-41.67%	8.33%
	Clarity	-0.38	0.32
Claude 3 Opus	Compliance	-9.09%	-8.33%
	Extensiveness	-6.84%	-13.57%
	Correctness	-25.76%	-30.00%
	Flow	-21.21%	-24.09%
	ExtraHyp	0.00%	0.00%
	ExtraAxiom	27.27%	25.00%
	ExtraROI	0.00%	0.00%
	Clarity	-0.57	-0.3
Gemma 2 (9B)	Compliance	0.00%	0.00%
	Extensiveness	-14.29%	2.38%
	Correctness	0.00%	0.00%
	Flow	-3.33%	0.00%
	ExtraHyp	-41.67%	8.33%
	ExtraAxiom	0.00%	0.00%
	ExtraROI	-8.33%	-8.33%
	Clarity	-4	0
Llama 3 (8B)	Compliance	33.33%	0.00%
	Extensiveness	-10.95%	-16.67%
	Correctness	-3.33%	3.33%
	Flow	-11.67%	-5.00%
	ExtraHyp	-25.00%	8.33%
	ExtraAxiom	-16.67%	83.33%
	ExtraROI	-33.33%	0.00%
	Clarity	-0.3	0.09
Llama 3.1 (8B)	Compliance	25.00%	-8.33%
	Extensiveness	-4.76%	-13.33%
	Correctness	0.00%	0.00%
	Flow	0.00%	0.00%
	ExtraHyp	-25.00%	-8.33%
	ExtraAxiom	16.67%	83.33%
	ExtraROI	-8.33%	8.33%
	Clarity	-0.25	0.58

Table 2: Difference in model performance, measured by our introduced metrics, when the model is presented with the perturbed input versus the original symbol (denoted as "→ to ♠"), and when the model is presented with the Selected Context versus the Full Context (denoted as "SC to FC"). Values in green denote an increase in performance and values in red denote a decrease in performance.

vide a proof). We can see that models like Claude 3 Opus and Gemma 2 were more consistent in the clarity of their outputs. Although we cannot guarantee that the gold standard proof is the only possible proof, we can see from Table 3 that Correctness and Flow align well with the clarity score, i.e., they all decrease as the clarity decreases. This indicates that as the model struggles to complete the proof, the output becomes more convoluted and difficult to understand.

Model	Clar.	Ext.	Corr.	Flow
GPT-3.5-turbo	0∼1	60.00%	60.00%	0.00%
	1∼2	57.14%	0.00%	0.00%
	2∼3	32.00%	0.00%	8.00%
	3∼4	49.64%	16.25%	33.75%
Claude 3 Opus	2∼3	60.00%	20.00%	20.00%
	3∼4	88.44%	66.36%	79.09%
Gemma 2 (9B)	3∼4	34.29%	0.00%	8.00%
Llama 3 (8B)	1∼2	41.43%	3.33%	0.00%
	2∼3	33.85%	4.62%	7.69%
	3∼4	67.62%	13.33%	40.00%
Llama 3.1 (8B)	0∼1	20.00%	0.00%	5.00%
	1∼2	37.50%	0.00%	2.50%
	2∼3	42.54%	4.44%	13.33%
	3∼4	25.71%	0.00%	0.00%

Table 3: Summary of our results grouping by Clarity value bins, where Clar., Ext. and Corr. are short for Clarity, Extensiveness and Correctness, respectively.

To better understand the impact of the different scenarios on models, we compute the difference in model performance when the model is presented with the perturbed input versus the original symbol and when the model is presented with the Selected Context versus the Full Context. For this study, in addition to the metrics introduced before, we also compute:

- The percentage of times the model uses a hypothesis that was not necessary (ExtraHyp), as per our proof of reference.
- The percentage of times the model uses an axiom that was not necessary (ExtraAxiom), as per our proof of reference.
- The percentage of times the model uses a rule of inference that was not provided in the input (ExtraROI).

Table 2 summarizes these findings. The introduction of a perturbation in the input had a bigger impact on Clarity than the change in context for all models. Looking at "→ to ♠" for both Llama models and ChatGPT, → has a higher percentage of steps, but also more unnecessary steps. The symbol replacement seems to be curbing all types of steps. For ChatGPT, we noticed that there is a bigger drop in Extensiveness for "SC to FC" than "→ to ♠", but the drop in Correctness and Flow is smaller for "SC to FC" than "→ to ♠". This suggests that ChatGPT struggles to retrieve the necessary axioms from the givens, but the change in context does not seem to significantly impact its Correctness. For

Claude, both changes seem equally challenging, with the change in context having slightly bigger impact on Extensiveness, Correctness, and Flow. Claude 3 Opus was the only model to have a decrease across all metrics for both differences.

There are three different types of steps that could appear in our studied proof: Hypotheses, Axioms, and Rules of Inference. With that in mind, we split each of our metrics into these three cases and summarize the results on Table 4. We noticed that models tend to be better at manipulating axioms than at listing hypotheses or applying rules of inference. In particular, both Llama models struggled with rules of inference (ROIs), with no Extensiveness value being higher than 25% and all Correctness values being 0%. For these two models we also noticed that while mistakes on substitution (Correctness) seem to lead to fewer contributions to the final proof (low Flow), mistakes on rules of inference still lead to contributions. We think this indicates that these models use rule of inference to justify steps they "know" are needed, even though the rule never correctly justifies said step.

We also observed that Gemma 2 never attempted to use an axiom (all values are 0%), and its best performance was on Extensiveness for hypotheses. ChatGPT had the biggest gap on Extensiveness for axioms (SC vs FC), and on the Full Context scenario, it always made mistakes on substitution. Claude 3 Opus was slightly worse at applying ROIs than making substitutions on easiest scenario (SC with $\rightarrow$), but it did much worse at substitution than on ROIs on the hardest scenario (FC with ♠). Its only use of unnecessary steps was on the hardest scenario and it used extra axioms.

We also analyze the spans highlighted by the annotators, alongside the comments left by them on each location. We collected a total of 304 annotations derived from highlighted spans, out of which, 170 were unique. We note that more than 99% of these comments denote errors made my the model, often related to the questions that were presented to the annotators. Based on this insight, we use this information to empirically estimate "when" models tend to make their first mistake as they generate the requested proof. Concretely, we compute the position in terms of number of characters, normalized by answer length, of the start of the spans highlighted by the annotator. These relative positions are then averaged for all the answers for each model.

As shown in Table 5, we see models tend to make their first mistake relatively early in their generation, which is true even for the more advanced black-box models. The table also shows how differently each model behaves in terms of verbosity, where we see models Llama 3 and 3.1 generating answers that are up to 3,000 characters long, while API-based models like GPT-3.5 and Claude 3 Opus are significantly more concise. The relatively lower value shown by Gemma is due to the model often not generating an actual proof, which artificially brings this number down. Compared to our gold standard proofs, which contain 478 and 452 characters, these results show that models tend to favor verbosity instead of precision when generating these kinds of proofs without specialized prompts.

Finally, we analyze the *content* of the annotations for each of the highlighted spans. We think these may provide additional insights on the nature of the mistakes by the models. In order to perform this analysis, we encode the annotations with Sentence-BERT (Reimers and Gurevych, 2019), via the *all-MiniLM-L6-v2* model, using the SentenceTransformers[1] package. We then perform clustering using k-means via its scikit-learn implementation (Pedregosa et al., 2011). We compute clusters varying parameter k, the number of clusters, with $k = 1, \ldots, 15$, and select the top 3 results as based on silhouette scores (Rousseeuw, 1987). Finally, we visually analyze these best results by performing PCA on the embedded annotation comments, and plotting them on a 2D chart, coloring the examples by cluster. We find the that $k = 7$ offers meaningful results, highlighting four distinct behaviors models often engage in the following cluster, which here we represent by the instance closest to the centroid: (Cluster 0) "*Inability to recognize the Unicode symbol used to replace the implication operator*", (Cluster 1) "*Does not follow from Axiom*", (Cluster 3) "*Substitution Issue*", (Cluster 5) "Incorrect usage of *modus ponens*".

Discussion It is known that if we have a theorem in propositional calculus, then it has arbitrarily many proofs. One cannot even guarantee there is a unique "minimal length" proof. Hence, one may question our comparison of model outputs against a fix proof. However, since we study changes, we need a baselines to compare against. This also makes it easier to evaluate the output systematically and consistently. We also note that if we

[1] github.com/UKPLab/sentence-transformers

Model	Cxt.	Hypotheses				Axioms								Rules of Inference							
		Ext.		ExtraHyp		Ext.		Corr.		Flow		ExtraAxiom		Ext.		Corr.		Flow		ExtraROI	
		→	♠	→	♠	→	♠	→	♠	→	♠	→	♠	→	♠	→	♠	→	♠	→	♠
GPT-3.5-turbo	SC	83%	67%	17%	17%	72%	69%	42%	14%	25%	14%	17%	0%	42%	28%	17%	0%	42%	28%	33%	0%
	FC	67%	100%	50%	0%	22%	6%	0%	0%	8%	0%	83%	100%	44%	31%	17%	11%	44%	31%	50%	0%
Claude 3 Opus	SC	100%	100%	0%	0%	92%	100%	92%	73%	92%	80%	0%	0%	94%	83%	83%	70%	94%	83%	0%	0%
	FC	100%	100%	0%	0%	75%	69%	44%	17%	75%	36%	0%	50%	83%	67%	83%	42%	83%	67%	0%	0%
Gemma 2 (9B)	SC	67%	0%	33%	0%	0%	0%	0%	0%	0%	0%	0%	0%	6%	0%	0%	0%	6%	0%	17%	0%
	FC	100%	0%	50%	0%	0%	0%	0%	0%	0%	0%	0%	0%	6%	0%	0%	0%	6%	0%	0%	0%
Llama 3 (8B)	SC	67%	33%	17%	0%	67%	83%	8%	8%	17%	8%	0%	0%	25%	6%	0%	0%	17%	6%	33%	17%
	FC	50%	33%	33%	0%	56%	42%	19%	8%	14%	0%	100%	67%	14%	0%	0%	0%	14%	0%	50%	0%
Llama 3.1 (8B)	SC	67%	0%	50%	0%	64%	86%	0%	8%	8%	6%	0%	0%	6%	6%	0%	0%	6%	6%	33%	0%
	FC	100%	0%	17%	17%	19%	50%	6%	0%	0%	0%	67%	100%	22%	11%	0%	0%	14%	11%	17%	33%

Table 4: Results disaggregated for different categories: Hypotheses, Axioms, and Rules of Inference. For the sake of presentation, values in this table are rounded to the closest integer.

Model	Answer Len.	Rel. Ann. Loc.
GPT-3.5 turbo	1,310 ± 500	15.573 % (10.34)
Claude 3 Opus	1,577 ± 594	14.126 % (10.18)
Gemma 2 (9B)	890 ± 43	28.247 % (11.90)
LLama 3 (8B)	2,181 ± 751	17.549 % (09.70)
Llama 3.1 (8B)	3,747 ± 5,024	33.592 % (21.69)

Table 5: Average length of model answers (**Answer Len.**), in characters, with their respective standard deviations, and average relative location of the comments left by the annotators (denoted as **Rel. Ann. Loc.**), normalized be answer length. For the latter, numbers in parenthesis show the standard deviation.

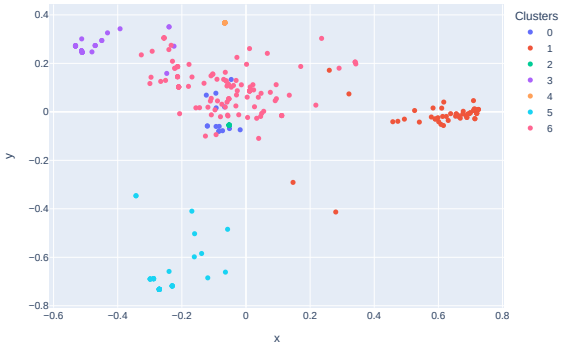

Figure 3: Clusters of annotations related to model behavior recognized as a mistake by our annotators.

had access to the training data, we could ensure we use the same proof used during training as our gold standard, but this is difficult or impossible in practice. To try to minimize this issue, we employ mathematicians to evaluate, which allow us to also collect comments about any conflict that could arise from this assumption. Finally, we would like to note that, ultimately, the difficulty of the model to follow the steps of the gold standard correlates with the decrease of overall clarity of the output.

5 Conclusions

This paper studies the generalization capabilities of LLM through the lens of mathematical reasoning. We perform an in-depth human evaluation of the output of LLMs when they are prompted to produce basic proofs in propositional calculus, comparing their answers when we replace the implication operator (→) with an unrelated, arbitrary symbol (♠). Our results show that nearly all our tested models produce lower quality proofs in this test, in particular open-weights models, suggesting the abilities of these LLMs to reason in this context have important limitations. For future work we would like to extend this study to incorporate more proofs, models, and multiple annotators. We would also like to analyze how models react to other input perturbations, for example using other replacement symbols, and/or alternative representations for them.

Limitations

While our study may provide valuable insights into the mathematical reasoning abilities of large language models, it is subject to several limitations. First, our analysis is constrained to a finite and small set of tasks, which do not capture the full breadth of mathematical reasoning scenarios. Second, human evaluation, while essential for assessing nuanced reasoning steps, is inherently subjective and may introduce variability in judgments. We would like to improve on this in future work by working with multiple annotators. Third, the models examined represent a snapshot of current architectures and training paradigms. Finally, our study focuses on English-language prompts leaving open questions about performance across languages.

References

Introducing Llama 3.1: Our most capable models to date. https://ai.meta.com/blog/meta-llama-3-1/.

AI@Meta. 2024. Llama 3 model card.

Miltiadis Allamanis, Pankajan Chanthirasegaran, Pushmeet Kohli, and Charles Sutton. 2017. Learning Continuous Semantic Representations of Symbolic Expressions. In *Proceedings of the 34th International Conference on Machine Learning*, pages 80–88. PMLR.

Anthropic. 2023. Introducing Claude. Anthropic.

Daniel G Bobrow. 1964. Natural language input for a computer problem solving system. *AI Technical Reports*.

Greg Brockman, Atty Eleti, Elie Georges, Joanne Jang, Logan Kilpatrick, Rachel Lim, Luke Miller, and Michelle Pokrass. 2023. Introducing ChatGPT and Whisper APIs. OpenAI Blog.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. QLoRA: Efficient finetuning of quantized llms. In *Advances in Neural Information Processing Systems*, volume 36, pages 10088–10115. Curran Associates, Inc.

Richard Evans, David Saxton, David Amos, Pushmeet Kohli, and Edward Grefenstette. 2018. Can Neural Networks Understand Logical Entailment? In *International Conference on Learning Representations*.

Edward A Feigenbaum and Julian Feldman. 1963. Computers and thought.

Simon Frieder, Martin Trimmel, Rashid Alawadhi, and Klaus Gy. 2023. LLM vs ITP. In *The 3rd Workshop on Mathematical Reasoning and AI at NeurIPS'23*.

Aaron Grattafiori and the Llama 3 Team. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

John Harrison, Josef Urban, and Freek Wiedijk. 2014. History of Interactive Theorem Proving. In *Handbook of the History of Logic*, volume 9, pages 135–214. Elsevier.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. 2023. GPQA: A Graduate-Level Google-Proof Q&A Benchmark. *Preprint*, arxiv:2311.12022.

Dino Rossegger. 2019. *Introduction to Mathematical Logic*.

Peter J. Rousseeuw. 1987. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20:53–65.

Gemma Team. 2024. Gemma 2: Improving Open Language Models at a Practical Size. *Preprint*, arXiv:2408.00118.

Sean Welleck, Jiacheng Liu, Ronan Le Bras, Hannaneh Hajishirzi, Yejin Choi, and Kyunghyun Cho. 2021. Naturalproofs: Mathematical theorem proving in natural language. In *Thirty-fifth Conference on Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks Track*.

Sean Welleck, Jiacheng Liu, Ximing Lu, Hannaneh Hajishirzi, and Yejin Choi. 2022. Naturalprover: Grounded mathematical proof generation with language models. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Chase B. Wrenn. 2025. Naturalistic Epistemology: Internet Encyclopedia of Philosophy.