The Promise and Limits of LLMs in Constructing Proofs and Hints for Logic Problems in Intelligent Tutoring Systems

Sutapa Dey Tithi, Arun Kumar Ramesh, Clara DiMarco, Xiaoyi Tian, Nazia Alam, Kimia Fazeli, Tiffany Barnes

Abstract

Intelligent tutoring systems have demonstrated effectiveness in teaching formal propositional logic proofs, but their reliance on template-based explanations limits their ability to provide personalized student feedback. While large language models (LLMs) offer promising capabilities for dynamic feedback generation, they risk producing hallucinations or pedagogically unsound explanations. We evaluated the stepwise accuracy of LLMs in constructing multi-step symbolic logic proofs, comparing six prompting techniques across four state-of-the-art LLMs on 358 propositional logic problems. Results show that DeepSeek-V3 achieved superior performance with 84.4% accuracy on stepwise proof construction and excelled particularly in simpler rules. We further used the best-performing LLM to generate explanatory hints for 1,050 unique student problem-solving states from a logic ITS and evaluated them on 4 criteria with both an LLM grader and human expert ratings on a 20% sample. Our analysis finds that LLM-generated hints were 75% accurate and rated highly by human evaluators on consistency and clarity, but did not perform as well explaining why the hint was provided or its larger context. Our results demonstrate that LLMs may be used to augment tutoring systems with logic tutoring hints, but requires additional modifications to ensure accuracy and pedagogical appropriateness.

Keywords: Intelligent Tutoring Systems, Next-step hints, Symbolic Logic, LLMs, Generative AI, Data Mining, DeepSeek-V3, GPT-40.

1. Introduction

Intelligent tutoring systems (ITSs) serve as scalable alternatives to human tutoring [20, 49]. These systems automatically evaluate student solutions and provide adaptive feedback, including next-step hints and explanations. Such personalized assistance has been shown to enhance student learning outcomes [50]. Although various data-driven methods have been shown to save time and resources by reducing the need for an expert to construct hints, they rely on the quantity and quality of training data. Moreover, the hints in these systems are often template-based with limited adaptation [45]. In this context, large language models (LLMs) can offer opportunities to augment ITS capabilities through scalable feedback generation [48]. However, integrating LLMs into ITSs faces challenges: LLMs may hallucinate incorrect information [52], which risks misleading students and reinforcing misconceptions [31]. LLMs may also generate correct answers without adhering to sound instructional principles, potentially limiting their effectiveness in fostering deep learning [22, 4]. Complicating matters for integrating LLM-based help in tutoring systems, novices may lack metacognitive skills to identify what and when to ask for help (help avoidance), and they could abuse the system by asking for direct answers or overly frequent hints (help abuse) [40]. Thus, the quality of solutions, adherence to pedagogical principles, and adaptation based on metacognitive needs are as crucial as accuracy for help in learning environments [18].

Formal logic proofs are essential in STEM education. Several studies have evaluated LLM performance in solving logic problems and handling natural language logical reasoning tasks [32, 41]. However, LLMs' ability to handle complex, multi-step symbolic logic proofs within pedagogical contexts is largely unexplored. In this work, we evaluate how well LLMs construct multi-step symbolic propositional logic proofs and next-step natural language hints in an intelligent logic tutor. We compared six prompting techniques across four state-of-the-art LLMs on 358 propositional logic problems, leveraging advanced prompt engineering to incorporate learning science theories. The quality of hints and explanations from the best-performing LLM was assessed by a rubric-based LLM grader and 20% were also rated by human experts on a 4-dimensional rubric. We investigated the following research questions:

RQ1 To what extent can LLMs construct accurate step-by-step solutions for propositional logic (PL) proofs within a logic tutor?

- RQ1a: How do prompting strategies influence proof construction performance?
- RQ1b: How does performance vary across a selection of four open-source and proprietary LLM models?

RQ2 How well does the most accurate LLM generate pedagogically viable hints and explanations?

- RQ2a: How does hint correctness vary across logic rules of differing complexity?
- RQ2b: How do domain experts assess the pedagogical quality of LLM-generated explanations in terms of consistency, clarity, justification, and subgoaling?

2. Related Work

Intelligent Tutoring Systems and Automated Feedback. ITSs have established their effectiveness in promoting student learning through personalization across multiple fields, including mathematics [6, 21, 37, 11], probability [43, 12], logic [26], and programming languages [6, 35, 17]. The evolution of these systems has led to various data-driven techniques for providing adaptive support. Murray et al. [29] demonstrated that data-driven assistance can efficiently generate hints while reducing expert involvement. This approach was further advanced by Barnes et al. through the Hint Factory, a method for generating next-step hints in propositional logic [8]. Further research has confirmed that both on-demand and unsolicited hints can significantly enhance student learning when appropriately implemented [8, 44, 26]. Despite the popularity and benefits of data-driven hints in ITSs, the current approaches have some limitations. Many systems provide immediate next-step hints without providing conceptual explanations [35, 38]. Marwan et al. showed that novices perceived hints with explanations as significantly more relevant and interpretable than those without explanations in a block-based programming environment, and hints without conceptual explanation can decrease students' trust in their helpfulness [27]. While incorporating textual explanations alongside procedural hints has been shown to be useful, generating these explanations traditionally requires extensive manual effort.

Large Language Models in Logic. Recent applications demonstrate LLMs' effectiveness in providing conversational support and personalized instruction across various educational contexts [51, 47, 39]. Aashish et al. showed students in an introductory programming course utilized a custom GPT-4 AI tool for assignment help, and students primarily asked AI assistance for conceptual understanding rather than complete solutions [13]. Systems like HypoCompass have demonstrated that LLM-augmented ITS can generate high-quality training materials and significantly improve student performance [25]. However, the application of LLMs to formal logic instruction presents unique challenges. While foundational datasets like ProofWriter [46] enable multi-hop proofs and FOLIO [15] introduces complex first-order logic (FOL) expressions, existing benchmarks remain narrowly scoped. ProntoQA [41] focuses on just one FOL rule (modus ponens). LogicBench [32] covers a larger set of inference rules, and they reveal critical performance gaps even in state-of-the-art models like GPT-4. Moreover, performance can decrease for complex operations, with accuracy dropping below 45% for proofs requiring three or more steps [42]. Prompt engineering techniques like Chainof-Thought (CoT) have improved LLMs' reasoning capabilities, but ensuring the faithfulness of generated reasoning chains remains challenging [24]. Careful prompt engineering is also important in ensuring the effective utilization of LLMs in educational contexts [45]. These approaches predominantly address LLM performance on natural language logic problems or isolated inference rules, with limited work exploring multi-step symbolic propositional logic proofs. Additionally, these LLMs are evaluated using traditional metrics such as accuracy and coverage [33, 34]; they are rarely assessed for the suitability of their explanations for educational contexts. Our work bridges these gaps by investigating LLMs' capabilities in multi-step symbolic logic proof construction and explanatory hint generation, with the goal of combining the pedagogical rigor of traditional ITS with the natural language capabilities of LLMs. We extend traditional evaluation metrics to include consistency (the what), clarity & relevance (the readability), justification (the why), and subgoaling (describing the why within a larger context), as described below, providing a more comprehensive assessment of LLM-generated hint quality in logic education.

3. Methodology

Proof Construction. We selected two datasets to evaluate LLMs' performance in proof construction: (1) The Symbolic Propositional Logic (SPL) dataset contains 310 multi-step problems. These problems were originally in natural language form, then Gottlieb et al. systematically translated them into symbolic PL representations [14]. (2) The logic tutor dataset (LT) contains 48 PL problems of varying difficulty. These problems are designed specifically for educational purposes, with each proof typically requiring 12-15 steps to complete. In both SPL and LT datasets, each problem contains the given premises and a conclusion, which are fed into LLMs using carefully engineered prompts. LLMs output the complete step-by-step proof along with a structured solution in JSON format.

Logic Tutor Problem Solving State (PSS) Dataset Extraction. To enable realistic testing of next-step hint generation, a dataset was extracted from interaction logs of all solution attempts to 18 problems each by 30 students using the logic tutor for homework in Fall 2024. Students were randomly selected from all CS majors at a US public university, whose 2021-22 graduating class demographics comprise 83% men and 17% women; with 58% white, 18.5% Asian, 3% Hispanic/Latin, 2% Black/African American, 9% other races. IRB approval was obtained, and only authorized researchers could access participant data. We developed an automated pipeline to convert the interaction logs into a textual representation of student problem solving states (PSS) (snapshots of problem solving after each step) (Figure 1). These PSS representations were fed into our final LLM framework using carefully engineered prompts to generate targeted next-step hints along with conceptual explanations for each PSS (Figure 1).

Intelligent Logic Tutor. Our logic tutor presents propositional logic (PL) problems to students in a graphical representation. Each problem shows a set of given statements at the top of the workspace and a conclusion to be derived at the bottom (Figure 2). Students solve the problems by iteratively deriving new logic statements until they arrive at the conclusion. The tutor divides the problems into three sections: pretest, training, and posttest. Students' prior knowledge is measured in the pretest section. The training section consists of five ordered levels with increasing difficulty. This is the only section where students can request and receive hints. Finally, students complete a posttest section consisting of six problems without any tutor assistance. The tutor uses a data-driven method to generate next-step

Student Progress

Final Conclusion: $J \vee K$ Derivation Steps:

- 1. $F \to (G \land \neg H)$, Given
- 2. $I \wedge F$, Given
- 3. $H \vee J$, Given
- 4. $\neg F \lor (G \land \neg H)$, Derived from (1) using Impl

LLM-Generated Hint

Hint: Derive F using Simp on line 2.

Explanation:

Line 2 $(I \wedge F)$ contains F, which is important for simplifying other statements. By deriving F from $I \wedge F$, we can use it to simplify line $4 (\neg F \vee (G \wedge \neg H))$ and move closer to the conclusion.

Tutor-Generated Hint

Hint: Click $I \wedge F$ and click on rule Simplification to derive F.

Figure 1: Example of a student problem-solving state and generated hints. The top box presents the student's derivation steps, while the bottom two boxes present the LLM- and tutor-generated hints.

hints with template-based explanations to guide students given their current progress.

3.1. Model Selection and Prompt Engineering

We evaluated four state-of-the-art LLMs: Llama-3-70B-Instruct [2], Gemini-Pro [7], GPT-40 [1], and DeepSeek-V3 [23]. We set the temperature parameter as 0.1 for all LLMs to limit the randomness and variability of responses and improve accuracy. We opted not to use the LLMs, such as of or DeepSeek-Reasoner, as they currently require significant computational resources and are not yet practical for deployment in a classroom setting. Each prompt is structured consistently (Figure 3) and is designed to incorporate evidence-based learning science principles [45]. Our prompt engineering framework implements six increasingly advanced techniques, building upon Zero-Shot (ZS) and Few-Shot (FS) prompting [10] with Chain-of-Thought (CoT) reasoning [53].

Zero Shot (ZS) Zero-shot prompting elicits step-by-step solutions and output expectations without providing any training examples. Thus, it relies on the model's inherent reasoning and problem-solving capabilities, testing

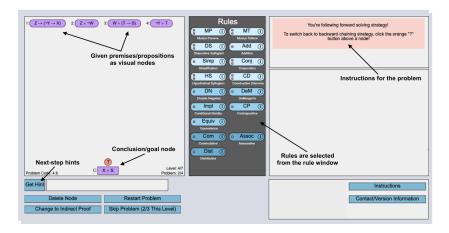


Figure 2: The full interface of the logic tutor, showing the given statements at the top of the workspace and a conclusion to be derived at the bottom. The middle pane contains the rule buttons.

its capacity to generalize with no prior guidance.

Few Shot Plan and Solve (FS_PlanAndSolve) This technique instructs the LLM to decompose a problem into subgoals and address each subgoal one by one. Two training examples are included in the prompt to demonstrate this approach, with the goal of helping the model learn to break its problem-solving process into smaller intermediate subgoals. This problem decomposition-recomposition strategy can promote logical consistency across complex tasks.

Few Shot Chain of Thought (FS_CoT) Chain-of-thought (CoT) prompting provides the model with examples that demonstrate a series of intermediate reasoning steps. Through these exemplars, the model may learn to articulate its own reasoning more transparently. Previous studies have shown that CoT significantly improves the reasoning abilities of LLMs [53].

Few Shot Logical Dual Chain of Thought (FS_L_DCoT) Building on the CoT concept, this approach integrates both direct and indirect proof strategies. This technique provides the LLM with examples that demonstrate how to consider alternate logical paths and cross-verify them. By encouraging the exploration of multiple reasoning paths, this approach reduces the likelihood of overlooking potential mistakes or alternative solutions.

Few Shot Bidirectional Logical Dual Chain of Thought (FS_BL_DCoT) This strategy augments the Dual CoT by combining forward

chaining (from premises to conclusion) with backward chaining (from conclusion back to premises). By working in both directions, the model may gain an additional layer of validation and error detection.

Few Shot Tree of Thoughts Chain of Thought (FS_ToT_CoT) In this approach, the solution path branches out across multiple reasoning "threads" or experts, each contributing a step-by-step proof. We added examples in the prompt where three experts collaboratively debated each step, acknowledged and corrected mistakes, and refined their step derivations. This collaborative interplay forms a tree-like structure of possible solution paths: if a line of reasoning fails, it is pruned when the expert acknowledges their mistake; successful lines continue to expand. This approach is meant to ensure that alternative proofs are considered and validated.

3.2. Data Splits

Each dataset is divided into training, validation, and test sets. Unlike a traditional machine learning setup, the training set consists of a hand-curated set of examples used for few-shot prompting. We iteratively refined the prompt templates based on the validation set performance until further modifications yielded diminishing returns. Once the prompt template was finalized, we ran the models only on the test set and reported the final results.

3.3. Evaluation Framework

The built-in logic checker in the logic tutor automatically verified the correctness of all LLM-generated proof steps and hints, ensuring relevance within the tutor context. We conducted a qualitative evaluation of LLMgenerated hints and explanations by two experts with 2-4 years of experience as teaching assistants and researchers for logic and ITS. The evaluation rubrics (Table 1) were adapted from the framework proposed by Roest et al. [39], including: consistency, clarity, justification, and subgoaling. To ensure their suitability for our tutor context, two domain experts collaborated and iteratively refined them until a consensus was reached with a course instructor expert. Twenty percent of the generated hints and explanations were rated on each rubric on a scale from 1 to 4 (with 4 being the best). We used Spearman correlation (ρ) and Quadratic Weighted Cohen's Kappa (QWK) to obtain inter-rater agreement of the assessment dimensions, which are common reliability measurements between two raters [48, 30]. We also used a rubric-based LLM grader to understand the potential of using it as an automated evaluator.

Abstract Prompt Template for Proof/Hint Generation

Role Assignement & Context

(e.g., "You are an intelligent tutor in propositional logic...")

Instructions

- 1. Step-by-step proofs, where each step should have derived node, parents, rule applied
- 3. Acceptable rules: <allowed rules in the tutor>
- 4. Final "structured solution" in JSON-like format

Output Expectation

- A detailed step-by-step derivation followed by a JSON-like "structured solution."
- A detailed step-by-step derivation, a next-step hint followed by a conceptual explanation

Few-Shot Examples

- Demonstrates prompt specific proof strategies, step-bystep derivations and the final JSON-like output.
- Demonstrates a step-by-step proof, one-line next-step hint, a conceptual explanation of the hint

User Prompt

References givens, conclusion and/or current progress and instructions to produce a formal proof/hint.

Figure 3: Abstract prompt template incorporating context, instructions, output expectations, examples, and user prompts.

4. Results

4.1. RQ1: Logic Proof Construction Performance

We evaluated the performance of four LLMs (Llama-3, Gemini-Pro, GPT-4o, and DeepSeek-V3) in six prompting techniques on the two logic tutor (LT) and symbolic propositional logic (SPL) datasets. DeepSeek-V3 demonstrated superior accuracy in most prompt configurations, with the best FS_ToT_CoT prompts achieving 85.0% and 86.7% accuracy of the steps generated for full solutions of LT and SPL problems, respectively (Table 2). GPT-4o also exhibited strong performance, particularly on SPL with FS_CoT prompts (83.6%). Gemini-Pro showed mixed results, while Llama-3 generally underperformed. All zero-shot prompted models underperformed,

Criteria	Definition
Consistency	Does the hint follow the tutor's domain rules and align with the student's current proof state?
Clarity	Is the hint clearly written and free of irrelevant or excessive details?
Justification	Does the hint provide a rationale or reasoning for why the next step was suggested?
Subgoaling	Does the hint provide a larger context (without giving away the full solution) explaining how the step will help work on other statements?

Table 1: Evaluation Criteria for LLM-generated Hint Explanations, adapted from Roest et al. [39]

indicating the importance of in-context prompting in the integration of LLMs into ITSs. Lengths of the parent statements were significantly longer for incorrect (M=7.01) versus correct (M=4.57) steps (t=-12.6, p<0.001). This suggests longer parent statements may introduce ambiguity, leading to more incorrect steps.

Prompt Technique	Llama-3		Gemini-Pro		GPT-4o		DeepSeek-V3	
	LT	SPL	LT	SPL	LT	SPL	LT	SPL
ZS	24.4	34.1	37.7	40.5	67.9	68.3	75.7	72.2
FS_CoT	65.7	62.6	59.0	67.0	76.9	83.6	83.7	84.4
FS_PlanAndSolve	56.7	60.2	49.8	50.3	75.5	79.9	79.0	83.0
FS_BL_DCoT	49.1	52.3	58.5	67.4	76.7	79.8	84.4	84.9
FS_L_DCoT	63.8	64.6	54.9	50.3	73.5	75.5	84.4	84.4
FS_ToT_CoT	59.1	56.1	51.7	60.4	72.7	82.4	85.0	86.7

Table 2: Performance of different LLMs in different zero-shot and few-shot settings, measured by accuracy on two datasets: LT and SPL.

Rule-Specific Performance Analysis. For all the FS-based prompting techniques and the GPT-40 and DeepSeek-V3 models, we calculated the average accuracy across different logical rules. As shown in Table 3, DeepSeek-V3 and GPT-40 both achieved high accuracy on basic rules (Com, Add, Conj, Simp) while DeepSeek-V3 maintained stronger performance on the rules MP,

DeM, MT, HS, DS and Contra. The performance of GPT-40 declined with higher complexity. The most challenging rules were Contra, CP, DN, and Impl, although DeepSeek-V3 still outperformed GPT-40.

Rule	Detail	DeepSeek-V3	GPT-40
Commutation	$(p \lor q) \implies (q \lor p)$	99.23	89.28
\mathbf{Add} ition	$p \implies (p \lor q)$	95.46	87.45
Simp lification	$(p \land q \implies p)$	92.16	83.34
Conjunction	$(p,q) \implies (p \land q)$	90.76	90.00
Modus Ponens (\mathbf{MP})	$((p \to q) \land p) \implies q$	89.17	82.26
DeMorgan's (\mathbf{DeM})	$\neg (p \land q) \implies (\neg p \lor \neg q)$	87.01	76.97
Modus Tollens (\mathbf{MT})	$((p \to q) \land \neg q) \implies \neg p$	86.80	74.68
Hyp. Syll. (HS)	$((p \to q) \land (q \to r) \implies (p \to r)$	86.35	75.83
Disj. Syll. (\mathbf{DS})	$((p \lor q) \land \neg p) \implies q$	78.06	62.31
Contradiction	$(p \land \neg p) \iff 0$	76.67	38.24
Implication	$(p \to q) \iff (\neg p \lor q)$	72.26	68.52
Double Negation (DN)	$\neg \neg p \implies p$	69.20	67.05
Contrapositive (CP)	$(p \to q) \implies (\neg q \to \neg p)$	64.07	45.77

Table 3: Rule accuracy for comparison of DeepSeek-V3 and GPT-40 across all prompting techniques.

4.2. RQ2a: Evaluating Next-step Hint Generation Performance

Based on RQ1, DeepSeek-V3 performed better than all other LLMs. While more complex prompting like bidirectional dual CoT and tree-ofthoughts CoT improved accuracy, their lengthier and more complex prompts appeared difficult to adapt for the hint generation task. FS CoT achieved comparable accuracy with simpler prompting, making it our preferred choice. Using DeepSeek-V3 with FS CoT, we generated next-step hints and conceptual explanations for the LT PSS dataset containing 1,050 unique student problem-solving states. Each state represents a snapshot of a student's progress, containing all steps completed at that point, and transformed from a graphical into a natural language-based form. DeepSeek-V3 with FS CoT was used to generate the next best step hint for each state. A modified version of the LT logic checker was used to evaluate correctness, with any hint considered to be incorrect if the step was logically incorrect, the suggested step was already present in the student solution, or if the parent steps were not yet derived. Overall, 75% of the resulting hints were rated as correct/accurate. Accuracy across the specific logical rules is shown in Figure 4. The model struggled most with complex rules like DS, Impl, Add, DN, MT, DeM, CP, and CD. With the exception of Addition, all of these rules handle negations, introducing complexity into the application of rules. Addition introduces a new variable into a proof, which adds another type of complexity.

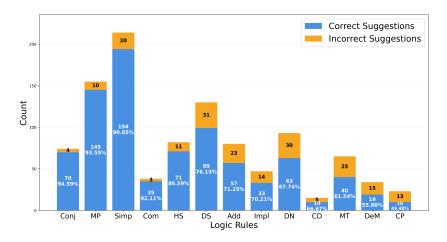


Figure 4: The distribution of correct and incorrect LLM-generated hints for each rule.

We compared the performance of DeekSeek-V3 to our logic tutor's data-driven hint mechanism, which utilizes historical data to identify optimal next steps. DeepSeek-V3 generated a higher number of unique hints per problem on average (M=4.5) compared to the tutor (M=3.0). The overall hint accuracy of 75% is promising. However, accuracy changed across the tutor levels. While the LLM achieved 87.9% accuracy in the first training level, the number of unique hints increased, and accuracy declined across the 5 training levels. Since difficulty and the number of logic rules needed increase at each level, the size of the potential solution space expands. This larger solution space may have led to lower hint accuracy and the generation of more unique hints by DeepSeek-V3.

4.3. RQ2b: Evaluating Next-step Hint Explanations

Two expert graders independently applied the evaluation rubrics (Table 1) to a sample of 240 LLM-generated conceptual explanations. Table 4 shows a strong inter-rater agreement for the expert human graders across all dimensions. The Spearman correlation (ρ) ranged from 0.77 to 0.93, and the QWK values ranged from 0.79 to 0.92, suggesting near-perfect reliability

Dimension	Human-Human		Avg Rating	Huma	an-LLM	Avg Rating	
	ρ	QWK	(Human)	ρ	QWK	(LLM)	
Consistency	0.90	0.85	3.9	0.59	0.30	3.81	
Clarity	0.93	0.92	3.09	0.54	0.56	2.92	
Justification	0.92	0.92	2.24	0.55	0.59	2.39	
Subgoaling	0.77	0.79	1.66	0.08^\dagger	0.16^\dagger	2.47	

Table 4: Inter-rater agreement metrics for evaluation of 210 LLM-generated explanations. $\rho = \text{Spearman's Correlation}, \ QWK = \text{Quadratic Cohen's Kappa}. \ ^{\dagger}p > \text{Bonferroni}_{\text{threshold}}.$

[28]. These high agreement metrics suggest that our evaluation criteria were well-defined and consistently applied by the expert human raters. The sampled LLM explanations were highly rated for consistency and clarity, with human-rater scores of 3.90 and 3.09 out of 4, respectively. However, the system showed room for improvement in other areas, with expert human rater scores of 2.24 for justification and 1.66 for subgoaling. We applied an LLM grader using the same rubrics to the full sample of explanations and compared their rubric scores with the expert human ratings. While the LLM grader assigned consistency and clarity scores comparable to human scores, it overestimated human scores in justification and subgoaling. However, the rubric-based LLM grader exhibited weaker agreement with human graders, particularly in subgoaling, where the average LLM subgoaling score was 2.47 as opposed to the human subgoaling score of 1.66.

5. Discussion

Logic rules can be considered basic or complex based on their use of negation. Basic rules such as Simplification or Commutative involve direct and single-step operations (e.g., $(p \lor q) \Longrightarrow (q \lor p)$). On the other hand, the rules involving negation (e.g., DeMorgans) cause challenges for people and for LLMs. The implication operator \to inherently includes negation as part of its definition $((p \to q) \iff (\neg p \lor q))$. Statements that combine both negation and implication increase the complexity even more, such as Modus Tollens and Contrapositive $((p \to q) \implies (\neg q \to \neg p))$. Johnson et al. argued that rules involving negation require working memory to track inverted truth values across multiple steps, thus increasing cognitive load [19]. Anderson et al. reported that humans exhibit slower response times

and higher error rates when processing negated statements like $\neg(A \land B)$, due to the increased cognitive load associated with maintaining dual truth state [5].

The challenge in tackling negation was also observed in the performance of the involved LLM models in our study. In logic proof construction, although DeepSeek-V3 (Table 2) achieved high accuracy on basic rules (e.g., Commutative (Com): 99.23%), the model struggled with more complex rules (e.g., CP: 64.07%). Our results suggest that LLMs encounter difficulties in propagating negation through inference chains, often struggling with expressions like $\neg(A \land B)$ and $\neg A \land \neg B$. This resembles a similar reasoning bottleneck found in human cognition.

The length of parent logic statements being combined also introduces complexity, usually via nested statements, which further introduces more cognitive load and working memory for humans. Similarly, a strong negative correlation was found between parent statement length and accuracy of LLMs. It may be that longer parent statements introduce ambiguities or combinatorial explosions that can exceed the contextual reasoning capacity of current LLM models. Preprocessing inputs into simpler structures may improve the logical reasoning performance of current LLMs.

Overall, DeepSeek-V3 with FS_CoT achieved an average accuracy of 75% for hint generation, but its hint accuracy was higher for earlier training problems and declined across the training levels as difficulty increased (from 87.9% to 63.2%). These accuracy levels are not high enough for integration into ITSs, as student trust can shift quickly when encountering even one that is inaccurate or not aligned with student thinking, potentially leading to help avoidance [36]. In our experiment, LLMs also sometimes generated sub-optimal hints based on the redundant statements derived by the students and failed to reorient the students to an optimal solution path.

Our LLM-generated hint explanations achieved higher scores on the consistency and clarity dimensions (Table 4), indicating that our selected LLM could follow specified rules and provide clear explanations. However, its justification and subgoaling scores were comparatively lower, suggesting LLMs sometimes struggle to articulate why a particular step was suggested within the larger problem or subgoal context. This limitation echoes critiques of LLMs as "stochastic parrots" [9] in educational contexts—they replicate surface-level patterns but lack intentional pedagogy, which is critical for learning [3]. Together with inaccuracy and these explanation ratings, our results highlight the need to build hybrid systems in which LLMs propose

candidate hints, but they are checked for domain validity (e.g. symbolic checkers in math [16]) and for pedagogical appropriateness.

6. Conclusion

Our work provides insights into the potential and current limitations of LLMs in logic tutoring systems, specifically examining their capabilities across proof construction, hint generation, and explanation quality. While models like DeepSeek-V3 and GPT-40 demonstrate promising performance in generating step-by-step solutions (achieving up to 86.7% accuracy) and producing factually correct hints (75% accuracy) with clear and consistent conceptual explanations, significant challenges remain in ensuring better pedagogical alignment. Our study also had several limitations. Our evaluation was limited to single-step accuracy for problem-solving in one domain. Future work should evaluate the optimality of LLM-constructed full solutions. The human evaluation of hint explanations covered only 20% of the dataset, potentially limiting generalizability; and although we evaluated 4 criteria for hint explanations, other pedagogical elements may be important for human understanding and learning. Future work should focus on developing hybrid architectures to combine LLM capabilities with learning science principles to bridge the gap between the potential of LLMs and the nuanced demands of a pedagogical environment.

References

- [1] Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F.L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., et al., 2023. Gpt-4 technical report. arXiv preprint arXiv:2303.08774.
- [2] AI@Meta, 2024. Llama 3 model card URL: https://github.com/meta-llama/llama3/blob/main/MODEL_CARD.md.
- [3] Aleven, V., Stahl, E., Schworm, S., Fischer, F., Wallace, R., 2003. Help seeking and help design in interactive learning environments. Review of educational research 73, 277–320.
- [4] Aleven, V.A., Koedinger, K.R., 2002. An effective metacognitive strategy: Learning by doing and explaining with a computer-based cognitive tutor. Cognitive science 26, 147–179.

- [5] Anderson, J.R., 2013. The adaptive character of thought. Psychology Press.
- [6] Anderson, J.R., Corbett, A.T., Koedinger, K.R., Pelletier, R., 1995. Cognitive tutors: Lessons learned. The journal of the learning sciences 4, 167–207.
- [7] Anil, R., Borgeaud, S., Wu, Y., Alayrac, J.B., Yu, J., Soricut, R., Schalkwyk, J., Dai, A.M., Hauth, A., Millican, K., et al., 2023. Gemini: A family of highly capable multimodal models. arXiv preprint arXiv:2312.11805 1.
- [8] Barnes, T., Stamper, J.C., Lehmann, L., Croy, M.J., 2008. A pilot study on logic proof tutoring using hints generated from historical student data., in: EDM, pp. 197–201.
- [9] Bender, E.M., Gebru, T., McMillan-Major, A., Shmitchell, S., 2021. On the dangers of stochastic parrots: Can language models be too big?, in: Proceedings of the 2021 ACM conference on fairness, accountability, and transparency, pp. 610–623.
- [10] Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J.D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al., 2020. Language models are few-shot learners. Advances in neural information processing systems 33, 1877–1901.
- [11] Canfield, W., 2001. Aleks: A web-based intelligent tutoring system. Mathematics and Computer Education 35, 152.
- [12] Chi, M., VanLehn, K., 2010. Meta-cognitive strategy instruction in intelligent tutoring systems: how, when, and why. Journal of Educational Technology & Society 13, 25–39.
- [13] Ghimire, A., Edwards, J., 2024. Coding with ai: How are tools like chatgpt being used by students in foundational programming courses, in: International Conference on Artificial Intelligence in Education, Springer. pp. 259–267.
- [14] Gottlieb, E., 2024. propositional-logic. URL: https://huggingface. co/datasets/ergotts/propositional-logic, doi:https://doi.org/ 10.5281/zenodo.14567918.

- [15] Han, S., Schoelkopf, H., Zhao, Y., Qi, Z., Riddell, M., Zhou, W., Coady, J., Peng, D., Qiao, Y., Benson, L., et al., 2022. Folio: Natural language reasoning with first-order logic. arXiv preprint arXiv:2209.00840.
- [16] He-Yueya, J., Poesia, G., Wang, R.E., Goodman, N.D., 2023. Solving math word problems by combining language models with symbolic solvers. arXiv preprint arXiv:2304.09102.
- [17] Holland, J., Mitrovic, A., Martin, B., 2009. J-latte: a constraint-based tutor for java.
- [18] Holstein, K., McLaren, B.M., Aleven, V., 2019. Designing for complementarity: Teacher and student needs for orchestration support in ai-enhanced classrooms, in: Artificial Intelligence in Education: 20th International Conference, AIED 2019, Chicago, IL, USA, June 25-29, 2019, Proceedings, Part I 20, Springer. pp. 157–171.
- [19] Johnson-Laird, P.N., 2001. Mental models and deduction. Trends in cognitive sciences 5, 434–442.
- [20] Kardan, S., Conati, C., 2015. Providing adaptive support in an interactive simulation for learning: An experimental evaluation, in: Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems, pp. 3671–3680.
- [21] Koedinger, K.R., Anderson, J.R., Hadley, W.H., Mark, M.A., et al., 1997. Intelligent tutoring goes to school in the big city. International Journal of Artificial Intelligence in Education 8, 30–43.
- [22] Koedinger, K.R., Corbett, A.T., Perfetti, C., 2012. The knowledge-learning-instruction framework: Bridging the science-practice chasm to enhance robust student learning. Cognitive science 36, 757–798.
- [23] Liu, A., Feng, B., Xue, B., Wang, B., Wu, B., Lu, C., Zhao, C., Deng, C., Zhang, C., Ruan, C., et al., 2024a. Deepseek-v3 technical report. arXiv preprint arXiv:2412.19437.
- [24] Liu, T., Xu, W., Huang, W., Wang, X., Wang, J., Yang, H., Li, J., 2024b. Logic-of-thought: Injecting logic into contexts for full reasoning in large language models. arXiv preprint arXiv:2409.17539 URL: https://arxiv.org/pdf/2409.17539.

- [25] Ma, Q., Shen, H., Koedinger, K., Wu, S.T., 2024. How to teach programming in the ai era? using llms as a teachable agent for debugging, in: International Conference on Artificial Intelligence in Education, Springer. pp. 265–279.
- [26] Maniktala, M., Barnes, T., 2020. Deep thought: An intelligent logic tutor for discrete math, in: Proceedings of the 51st ACM Technical Symposium on Computer Science Education, pp. 1418–1418.
- [27] Marwan, S., Lytle, N., Williams, J.J., Price, T., 2019. The impact of adding textual explanations to next-step hints in a novice programming environment, in: Proceedings of the 2019 ACM conference on innovation and technology in computer science education, pp. 520–526.
- [28] McHugh, M.L., 2012. Interrater reliability: the kappa statistic. Biochemia medica 22, 276–282.
- [29] Murray, T., 2003. An overview of intelligent tutoring system authoring tools: Updated analysis of the state of the art. Authoring Tools for Advanced Technology Learning Environments: Toward Cost-Effective Adaptive, Interactive and Intelligent Educational Software, 491–544.
- [30] Naismith, B., Mulcaire, P., Burstein, J., 2023. Automated evaluation of written discourse coherence using gpt-4, in: Proceedings of the 18th Workshop on Innovative Use of NLP for Building Educational Applications (BEA 2023), pp. 394–403.
- [31] Nye, B.D., Mee, D., Core, M.G., 2023. Generative large language models for dialog-based tutoring: An early consideration of opportunities and concerns., in: LLM@ AIED, pp. 78–88.
- [32] Parmar, M., Patel, N., Varshney, N., Nakamura, M., Luo, M., Mashetty, S., Mitra, A., Baral, C., 2024. Logicbench: Towards systematic evaluation of logical reasoning ability of large language models, in: Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 13679–13707.
- [33] Phung, T., Cambronero, J., Gulwani, S., Kohn, T., Majumdar, R., Singla, A., Soares, G., 2023. Generating high-precision feedback for programming syntax errors using large language models. arXiv preprint arXiv:2302.04662.

- [34] Phung, T., Pădurean, V.A., Singh, A., Brooks, C., Cambronero, J., Gulwani, S., Singla, A., Soares, G., 2024. Automating human tutor-style programming feedback: Leveraging gpt-4 tutor model for hint generation and gpt-3.5 student model for hint validation, in: Proceedings of the 14th Learning Analytics and Knowledge Conference, pp. 12–23.
- [35] Price, T.W., Dong, Y., Lipovac, D., 2017a. isnap: towards intelligent tutoring in novice programming environments, in: Proceedings of the 2017 ACM SIGCSE Technical Symposium on computer science education, pp. 483–488.
- [36] Price, T.W., Zhi, R., Barnes, T., 2017b. Hint generation under uncertainty: The effect of hint quality on help-seeking behavior, in: Artificial Intelligence in Education: 18th International Conference, AIED 2017, Wuhan, China, June 28–July 1, 2017, Proceedings 18, Springer. pp. 311–322.
- [37] Ritter, S., Anderson, J.R., Koedinger, K.R., Corbett, A., 2007. Cognitive tutor: Applied research in mathematics education. Psychonomic bulletin & review 14, 249–255.
- [38] Rivers, K., Koedinger, K.R., 2017. Data-driven hint generation in vast solution spaces: a self-improving python programming tutor. International Journal of Artificial Intelligence in Education 27, 37–64.
- [39] Roest, L., Keuning, H., Jeuring, J., 2024. Next-step hint generation for introductory programming using large language models, in: Proceedings of the 26th Australasian Computing Education Conference, pp. 144–153. URL: https://arxiv.org/pdf/2312.10055.
- [40] Roll, I., Aleven, V., McLaren, B.M., Koedinger, K.R., 2011. Improving students' help-seeking skills using metacognitive feedback in an intelligent tutoring system. Learning and instruction 21, 267–280.
- [41] Saparov, A., He, H., 2022. Language models are greedy reasoners: A systematic formal analysis of chain-of-thought. arXiv preprint arXiv:2210.01240.
- [42] Saparov, A., Pang, R.Y., Padmakumar, V., Joshi, N., Kazemi, M., Kim, N., He, H., 2023. Testing the general deductive reasoning capacity of

- large language models using ood examples. Advances in Neural Information Processing Systems 36, 3083–3105.
- [43] Sedlmeier, P., 1997. Basicbayes: A tutor system for simple bayesian inference. Behavior Research Methods, Instruments, & Computers 29, 328–336.
- [44] Stamper, J., Eagle, M., Barnes, T., Croy, M., 2013. Experimental evaluation of automatic hint generation for a logic tutor. International Journal of Artificial Intelligence in Education 22, 3–17.
- [45] Stamper, J., Xiao, R., Hou, X., 2024. Enhancing llm-based feedback: Insights from intelligent tutoring systems and the learning sciences, in: International Conference on Artificial Intelligence in Education, Springer. pp. 32–43.
- [46] Tafjord, O., Mishra, B.D., Clark, P., 2020. Proofwriter: Generating implications, proofs, and abductive statements over natural language. arXiv preprint arXiv:2012.13048.
- [47] Taneja, K., Maiti, P., Kakar, S., Guruprasad, P., Rao, S., Goel, A.K., 2024. Jill watson: A virtual teaching assistant powered by chatgpt, in: International Conference on Artificial Intelligence in Education, Springer. pp. 324–337.
- [48] Tian, X., Mannekote, A., Solomon, C.E., Song, Y., Wise, C.F., Mcklin, T., Barrett, J., Boyer, K.E., Israel, M., 2024. Examining llm prompting strategies for automatic evaluation of learner-created computational artifacts, in: Proceedings of the 17th International Conference on Educational Data Mining, pp. 698–706.
- [49] Ueno, M., Miyazawa, Y., 2017. Irt-based adaptive hints to scaffold learning in programming. IEEE Transactions on Learning Technologies 11, 415–428.
- [50] Van Lehn, K., 2011. The relative effectiveness of human tutoring, intelligent tutoring systems, and other tutoring systems. Educational psychologist 46, 197–221.
- [51] Venugopalan, D., Yan, Z., Borchers, C., Lin, J., Aleven, V., 2024. Combining large language models with tutoring system intelligence: A case

- study in caregiver homework support. arXiv preprint arXiv:2412.11995 URL: https://arxiv.org/pdf/2412.11995.
- [52] Wang, S., Xu, T., Li, H., Zhang, C., Liang, J., Tang, J., Yu, P.S., Wen, Q., 2024. Large language models for education: A survey and outlook. arXiv preprint arXiv:2403.18105.
- [53] Wei, J., Wang, X., Schuurmans, D., Bosma, M., Xia, F., Chi, E., Le, Q.V., Zhou, D., et al., 2022. Chain-of-thought prompting elicits reasoning in large language models. Advances in neural information processing systems 35, 24824–24837.