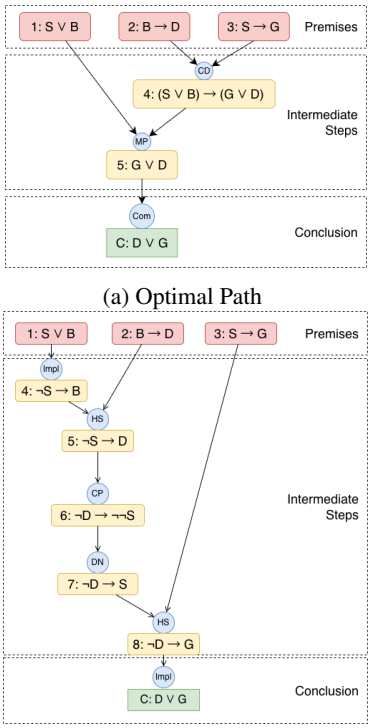
Confirming Correct, Missing the Rest: LLM Tutoring Agents Struggle Where Feedback Matters Most

Tahreem Yasir **Wenbo Li** **Sam Gilson**
Sutapa Dey Tithi Xiaoyi Tian Tiffany Barnes
North Carolina State University
{tyasir, wli55, sagilson, stithi, xtian9, tmbarnes}@ncsu.edu

Abstract

Effective tutoring requires distinguishing optimal, valid but suboptimal, and incorrect student solutions, a distinction central to intelligent tutoring systems (ITS) but untested for LLM-based tutors. As LLMs are increasingly explored as conversational complements to ITS, evaluating their diagnostic precision is essential. We present a benchmark of seven LLM feedback agents in propositional logic using knowledge-graph-derived ground truth across 10,836 solution–feedback pairs and three feedback conditions[1]. Models achieved near-ceiling performance on optimal steps but systematically over-rejected valid but suboptimal reasoning and over-validated incorrect solutions, precisely where adaptive tutoring matters most. These failures persisted across models regardless of solution context, suggesting architectural rather than informational limits. Moreover, accurate diagnosis did not reliably produce pedagogically actionable feedback, revealing a gap between diagnostic judgment and instructional effectiveness. Our findings suggest that LLMs are better suited for hybrid architectures where KG-grounded models handle diagnosis while LLMs support open-ended scaffolding and dialogue.



(a) Optimal Path

(b) Valid-Alternative Path

Figure 1: Optimal and valid-alternative solutions *(blue nodes represent abbreviated inference rule names, explained in* Table 4)

1 Introduction

Effective tutoring requires recognizing not just whether a student solution is correct, but whether it shows good reasoning that should be encouraged or guided toward a more efficient approach. (Gupta et al., 2025). In structured reasoning domains like propositional logic, students may produce optimal next steps that follow expert authored paths to conclusion, apply valid but suboptimal inference rules (*valid-alternative* steps), or make errors entirely (Maniktala et al., 2023). Examples of expert authored (optimal) solution and a valid but longer solution is illustrated in Figure 1.

Distinguishing among these three types of student steps has direct pedagogical implications identified as *assistance dilemma* by Learning Science experts (Koedinger and Aleven, 2007). Treating valid reasoning as incorrect can discourage productive exploration, while accepting all valid alternative steps without guidance can reinforce inefficient problem solving strategies. Intelligent Tutoring Systems (ITS) address this trade-off through step-level diagnosis of student reasoning using explicitly modeled solution spaces (VanLehn, 2011). However, they rely on expert-authored solution graphs for each problem, limiting scalability to new domains (Weitekamp et al., 2020), and provide limited support for conversational and exploratory dia-

[1] https://github.com/tahreemm/BEA_2026

Proceedings of the 21st Workshop on Innovative Use of NLP for Building Educational Applications, pages 819–840

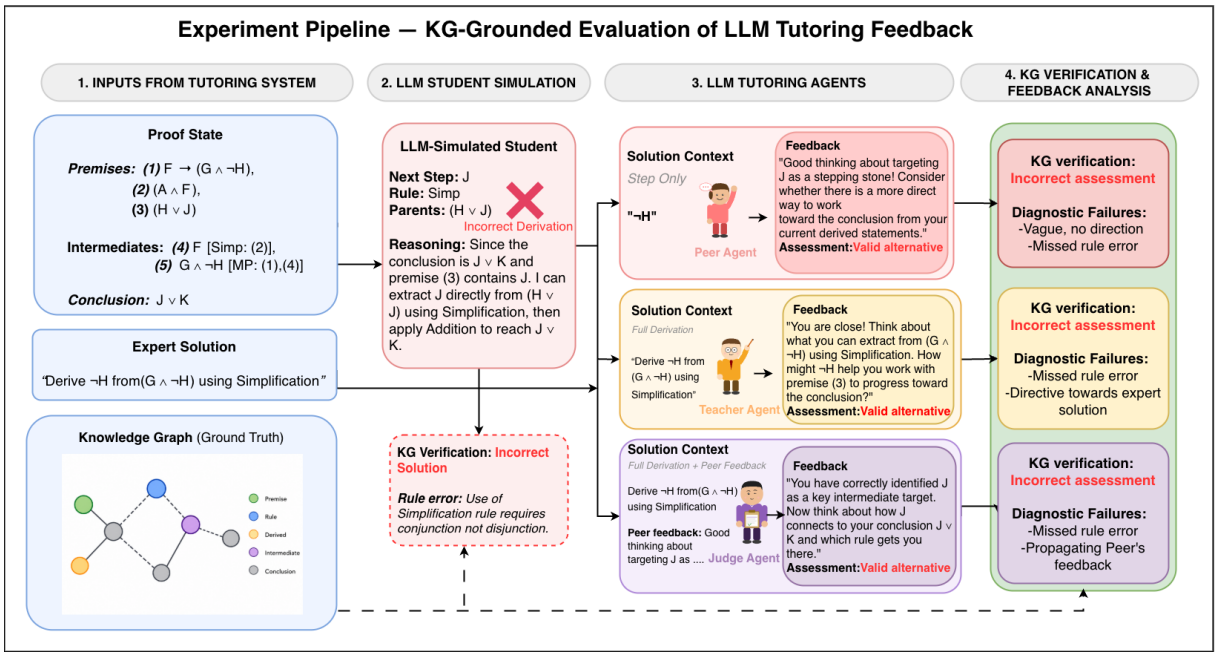


Figure 2: KG-grounded evaluation pipeline. (1) Proof state and expert solution are extracted from tutoring system. (2) LLM-simulated student produces an incorrect derivation (KG verification: Incorrect). (3) Three feedback agents assess student solution as "Valid-Alternative" incorrectly. (4) Automated and manual feedback analysis illustrates systematic over-validation and distinct diagnostic failure modes across Peer, Teacher, and Judge conditions.

logue when students diverge from expected reasoning paths (Zerkouk and Chikhaoui, 2025).

Large language models (LLMs) address these limitations through conversational flexibility, cross-domain generalization, and scalable feedback without per-problem authoring (Reddig et al., 2025; Chen et al., 2025), making them promising complements to ITS. However, without grounding, LLMs hallucinate (Liu et al., 2025a) and often reveal solutions instead of scaffolding reasoning (Macina et al., 2023), limiting their reliability for tutoring. Role-specialized feedback pipelines grounded in expert solutions may improve diagnostic accuracy (Phung et al., 2024; Guo et al., 2024), but whether they support the fine-grained reasoning diagnosis required for effective tutoring remains unclear, especially in structured reasoning domains.

We address this gap through a large-scale evaluation of LLM feedback in propositional logic, where distinguishing between optimal, valid-alternative, and incorrect steps has direct pedagogical importance. Unlike prior tutoring evaluations that rely on binary correctness labels (Gupta et al., 2025; Borchers and Shou, 2025), our framework uses a knowledge-graph (KG) solution space derived from a real tutoring system to represent all valid inference paths (Barnes and Stamper, 2008), enabling grounded three-way diagnosis central to intelligent tutoring systems (Aleven et al., 2009; VanLehn, 2006).

Contributions: We present the first benchmark for step-level evaluation of LLM tutoring feedback across multiple valid solution paths. Using this benchmark, we evaluate 10,836 LLM-simulated next-step solutions and their feedback with KG-grounded evaluation across seven models and three feedback conditions. To understand when LLM feedback remains pedagogically effective, and what factors influence this behavior, we study three research questions:

– *RQ1:* How accurately LLM feedback agents classify single-step logic proof solutions as optimal, valid-alternative, or incorrect, and what systematic errors emerge?

– *RQ2:* What model- and problem-level factors affect this three-way diagnosis?

– *RQ3:* What is the pedagogical quality of LLM feedback, and how it relates to three-way diagnosis?

Our fine-grained evaluation shows that LLMs consistently over-reject valid-alternative reasoning and over-validate incorrect solutions across models and solution contexts, with failures driven more by model behavior than problem difficulty. We further find that accurate solution diagnosis does not reliably produce pedagogically effective feed-

back. Models may correctly assess solution quality yet still fail to provide guidance that supports learning, reducing feedback to confirmation or error detection manifesting *assistance dilemma*. Together, these findings suggest that current LLMs require ITS-grounded diagnostic mechanisms and are better suited to complement, rather than replace, adaptive tutoring systems.

2 Related Work

2.1 LLM Tutoring and Diagnostic Adaptivity

LLMs show promise for diagnosing student errors (Reddig et al., 2025) and supporting self-regulation (Chen et al., 2025), but tutoring evaluations show they cannot reliably identify reasoning errors, even with reference solutions (Liu et al., 2025b; Jia et al., 2024). Role-specialized feedback pipelines, where one agent generates feedback and another verifies it, can improve precision and reduce over-praise (Phung et al., 2024; Guo et al., 2024), yet verification may also propagate existing errors depending on response type (Guo et al., 2024; Yasir et al., 2026). Prior work further shows that LLMs exhibit limited adaptivity to tutoring context (Borchers and Shou, 2025) and struggle with partially correct responses where guidance is most needed (Mahdavi et al., 2025).

Multiple valid solution paths are common in structured reasoning domains like propositional logic (Große and Renkl, 2006; Maniktala et al., 2023). Traditional ITS address this using manually authored solution graphs that encode both optimal and valid-alternative paths (Aleven et al., 2009), but this limits scalability to pre-authored problems. Whether LLM tutoring agents can similarly perform three-way diagnosis of optimal, valid-alternative, and incorrect responses remains untested and is the central question of this work.

2.2 LLM-Simulated Students

Evaluating tutoring feedback requires student solutions covering optimal, valid-alternative, and incorrect responses, which are often underrepresented in real interaction logs (Maniktala et al., 2023). LLM-simulated students have emerged as a practical solution to this coverage problem. Prior work shows that LLMs can reliably simulate learner actions in open-ended environments (Mannekote et al., 2025), generate responses for evaluating programming tutoring systems (Phung et al., 2024), and produce multiple-choice answers aligned with real student profiles and responses (Lu and Wang, 2024). LLMs have also been shown to simulate students of different skill levels across educational domains (Benedetto et al., 2024), while generating realistic uncertainty, confusion, and mistakes that effectively challenge teacher agents in benchmarking settings (Shi et al., 2025).

In our work, simulation is additionally required because the tutoring system logs lack reasoning traces, making real interaction data insufficient for step-level reasoning evaluation.

2.3 Evaluation Benchmarks

Existing logic reasoning benchmarks do not support the step-level, multi-path evaluation required in our framework. ProofWriter (Tafjord et al., 2020) evaluates only final proof validity, obscuring step-level rule application. FOLIO (Han et al., 2024) provides expert-authored first-order logic problems but penalizes valid alternative derivations, while ProntoQA (Saparov and He, 2023) supports controlled evaluation with synthetic reasoning chains but cannot represent branching points where multiple inference rules may apply. LogicLearner (Inamdar et al., 2025) focuses on guided practice for basic logic problems, and its benchmark data is unavailable. To the best of our knowledge, none of these benchmarks support evaluating feedback quality for intermediate steps when multiple valid solutions exist.

2.4 Reasoning Difficulty in Logic Proofs

Structural complexity is a known challenge in propositional reasoning (Barnes and Stamper, 2008). Nested connectives increase cognitive load and make it harder to extract propositions needed for further derivation, reducing performance as rule complexity increases (Johnson-Laird and Wason, 1970). In logic proof tutoring, students take more time and derive more unnecessary propositions on steps with nested expressions (Shabrina et al., 2024). Structural nesting is also a strong predictor of LLM error in propositional proof construction, with incorrect steps associated with longer and more complex parent statements across models and prompting conditions (Tithi et al., 2025).

Proof step position introduces an additional challenge. Early steps require reasoning over many valid continuations, while later steps limit the remaining derivations (Shabrina et al., 2024). We therefore examine whether diagnostic failures are

driven by step complexity, proof position, or model-specific factors (Section 4).

3 Dataset and Knowledge Graph

3.1 Task Formulation

A propositional logic proof problem is defined as a tuple $(\mathcal{P}, C)$, where $\mathcal{P} = p_1, \ldots, p_n$ is the set of premises (statements assumed to be true) and C is the target conclusion. A proof state $\sigma = (\mathcal{P}, I, C)$ represents progress through the proof, where I is the ordered set of intermediate statements derived by applying inference rules $r \in R$ to statements in $\mathcal{P} \cup I$. Initially, $I = \emptyset$. A complete list of inference rules is provided in Table 4, Appendix A.

In this study, we investigate single-step prediction: deriving the optimal next step (statement) from $\mathcal{P} \cup I$ using exactly one inference rule application that minimizes the remaining derivation distance to C. We restrict the task to single-step prediction and the corresponding feedback generation and evaluation.

3.2 Dataset

To generate single step solution-feedback pairs, we use 516 unique proof states extracted from interaction logs of a propositional logic tutoring system deployed in an undergraduate Discrete Mathematics course at a large U.S. university (Barnes and Stamper, 2008) in Spring 2023. States are drawn from 32 proof problems across five practice levels of varying difficulty (introductory through expert), that provide context specific and on-demand hints we call solution context . Levels 1 (pre-test) and 7 (post-test) are excluded because they do not provide hints. Each instance is unique at triple level $(\mathcal{P}, I, C)$, to ensure a distinct proof state. An example proof state is illustrated in Figure 2 and Figure 7. Table 5, Appendix B summarizes data distribution across tutor difficulty levels, tutoring system illustrations are present in Appendix C.

Step Complexity: To further characterize the reasoning difficulty of proof states in our dataset, we also present step complexity, computed as a weighted sum of operators and nesting structure in the derived expression. Details on computing complexity are available in (Appendix E.1). In our dataset, complexity values ranges from 0 (atomic expressions with no logical operators) to 11 ($M = 3.07$, $SD = 2.5$), covering simple single-operator next steps through deeply nested compound expressions.

3.3 Knowledge Graph Construction

To provide ground truth that captures all logically valid solution paths in our dataset, we encode the complete solution space for each of the 32 problems as a knowledge graph $K = (\Sigma, E)$. Σ is the set of nodes representing all reachable proof states while E comprise the set of edges representing a valid single step inference between two proof states $(\sigma_i, \sigma_j) \in E$. The proof state σ_j is obtained from σ_i by deriving a new statement $\hat{s}$ via rule $r \in R$ applied to parent statements from $\mathcal{P} \cup I_i$, such that $I_j = I_i \cup \{\hat{s}\}$. The graph is rooted at $\sigma_0 = (\mathcal{P}, \emptyset, C)$ and terminates at σ^* where $C \in I^*$. Given a finite premise set and closed rule set R of 15 propositional inference rules, all reachable derivations are enumerable via forward chaining, making this encoding exhaustive by construction (Barnes and Stamper, 2008).

Each edge $(\sigma_i, \sigma_j) \in E$ in the KG is annotated with the inference rule and respective parent statements in σ_i used to derive the next step $\hat{s}$ in σ_j. This annotation is necessary because a predicted step $\hat{s}$ may be symbolically valid yet inferentially unjustified if attributed to an incorrect rule or unsupported parent statements, illustrated in Figure 2 *(student reasoning error)*.

Example: a model predicting C via Modus Ponens must correctly identify both $(A \rightarrow C)$ and A as parents; a prediction that produces C but misattributes the rule or parent statements is flagged as incorrect regardless of symbolic validity. Edge-level annotation thus enables verification of both what was derived and how the derivation is justified.

Distance to conclusion: is defined as the minimum number of derivation steps from the current proof state to the conclusion such that $d(\sigma) = \min_{\pi:\sigma \rightsquigarrow \sigma^*} |\pi|$, and computed via breadth-first search over K. In our dataset it ranges from [X] to [Y] ($M = Z$, $SD = W$), covering both early proof states with many remaining steps and late proof states close to the conclusion. Unlike prior benchmarks relying on binary correctness, our KG encodes all valid inference paths, enabling three-way diagnosis. A next step, $\hat{s}$ from state σ_t is classified as one of three categories as follows:

Optimal: following an expert-authored path, if $(\sigma_t, \hat{\sigma}_{t+1}) \in E$ and $d(\hat{\sigma}_{t+1}) = d(\sigma_t) - 1$: is a valid inference that strictly reduces distance to the conclusion.

Valid-alternative: if $(\sigma_t, \hat{\sigma}_{t+1}) \in E$ but

$d(\hat{\sigma}_{t+1}) > d(\hat{\sigma}_{t-1})$, that means it does not reduce distance to the conclusion.

Incorrect: $(\sigma_t, \hat{\sigma}_{t+1}) \notin E$, meaning $\hat{s}$ not derivable under any valid inference rule per K.

4 Methods

4.1 Experimental Pipeline

The experimental pipeline extends the framework established in (Yasir et al., 2026) by introducing three-way KG-grounded diagnosis, distinguishing valid-alternative from incorrect solutions, and analyzing over-rejection and over-validation as pedagogically distinct failure modes. We construct next-step solution-feedback pairs using LLM-simulated next-step solutions (Section 4.2) generated from proof states in our dataset (Section 3.2), paired with corresponding LLM feedback under three prompt-level role-specialized feedback conditions (Section 4.3). Figure 1 summarizes this pipeline.

We evaluate seven LLMs, including reasoning-augmented models (GPT-o3, DeepSeek-R1, Qwen-3-32B) and instruction-tuned models (GPT-4.1, Gemini-1.5-Pro, LLaMA-3.3-70B, Mistral-Large). The set includes both proprietary and open-weight models, enabling comparison across capability levels relevant to educational deployment. Each model serves in two roles: (1) as student simulator generating solutions for all 516 proof states and (2) feedback agent evaluating all solutions under the three feedback conditions (Section 4.3). This produce $516 \times 7 \times 3 = 10{,}836$ solution-feedback pairs (516 proof states, seven LLMs, and three feedback conditions). This design, avoids bias from mismatched model pairs and enable model-level analysis of solution diagnosis and feedback quality across feedback conditions, directly addressing RQ1 and RQ2. Each pair represents one diagnosis instance: a next-step solution and reasoning trace generated by a model and evaluated under one feedback condition. We reuse the same simulated solutions across conditions for each model to ensure fair comparison.

4.2 Student Simulation

The tutoring system logs record only the predicted next step, inference rule, and parent statements, but not the reasoning trace. Without it, evaluating whether LLM feedback agents assess reasoning validity in addition to symbolic correctness is not possible. To address this, we use LLM simulation to generate solutions with reasoning traces. LLM-based simulation is an established methodology for tutoring evaluation (Mannekote et al., 2025; Phung et al., 2024), and prior work shows that simulated responses contain realistic confusion and mistakes suitable for benchmarking teacher models (Shi et al., 2025).

The student simulator prompt follows the design established in (Yasir et al., 2026). Each model receives a proof state $\sigma = (\mathcal{P}, I, C)$ and generates: (1) a predicted next step in symbolic form, (2) the inference rule used, (3) parent statements, and (4) a natural language reasoning trace. The complete prompt is shown in Figure 14. Models generate the most appropriate next step for the current proof state without instruction to produce a specific solution category. The solution category (optimal, valid-alternative, or incorrect) is later determined by comparison with the KG (Section 3.3). Error patterns observed during human evaluation for response annotation align with common student errors in the tutoring logs, including rule misapplication and incorrect parent selection.

4.3 Feedback Conditions

Accurate diagnosis alone is insufficient for effective tutoring and must be translated into pedagogically appropriate feedback. **Optimal** steps may require only confirmation and encouragement to continue, **Valid-alternative** steps require acknowledging correct reasoning while guiding toward a more efficient approach, and **Incorrect** steps require error identification and corrective guidance (VanLehn, 2006).

To evaluate whether LLM feedback agents can meet these requirements, and whether performance depends on solution context, we define three prompt-level feedback conditions that differ only in the expert solution information provided to the agent, keeping the task and other constraints constant. The three feedback conditions and their prompts are adopted from (Yasir et al., 2026). Each condition represents a distinct tutoring role:

– **Peer** receives only the next step in symbolic form, representing a peer who knows the answer but lacks effective reasoning. This condition tests whether correctness-only grounding is sufficient for accurate diagnosis and pedagogically appropriate feedback.

– **Teacher** represents a real life teacher with complete solution, including the inference rule and parent statements. This tests whether full derivational context improves diagnostic accuracy and feedback quality beyond the Peer condition.

– Judge receives the complete solution (inference rule and parent statements) along with the Peer's feedback, representing a teacher reviewing a teaching assistant's feedback. This setup is consistent with evaluation and verification loops in ITS (Rodrigues et al., 2025) and aligns with the LLM-as-Judge paradigm (Zheng et al., 2023). This condition tests whether downstream verification can correct upstream diagnostic errors and improve pedagogical feedback quality. Figure 1 shows the information available in each feedback agent.

All feedback agents are prompt-level conditions rather than distinct model architectures. Pedagogical constraints were kept constant across conditions: agents were instructed to scaffold reasoning without revealing answers, acknowledge correct reasoning before addressing errors, and limit feedback to 2–3 sentences (VanLehn, 2006). This design isolates information access as the main independent variable, enabling direct comparison of whether richer solution context improves diagnostic accuracy (*RQ1*) and feedback quality (*RQ3*). Prompt templates are provided in Appendix J.

5 Evaluation

5.1 Automated Metrics

To address RQ1 and RQ2, we compute three-way diagnostic performance and identify factors governing misdiagnosis using automated metrics derived from our KG ground truth.

Classification Performance (RQ1) Each feedback agent is prompted to label the solution as Optimal, Valid-alternative, or Incorrect. We compare these labels with KG-derived ground truth to compute F1 scores. However, F1 alone obscures pedagogically distinct failure modes: labeling valid-alternative reasoning as incorrect discourages productive exploration, while labeling incorrect solutions as valid reinforces misconceptions. To capture these two horns of the assistance dilemma, we define two conditional error rates:

Over-rejection (OR): is the rate at which valid-alternative solutions are incorrectly labeled as incorrect, discouraging productive reasoning.

Over-validation (OV): is the rate at which incorrect solutions are labeled as valid, reinforcing misconceptions.

Factors affecting misclassification (RQ2) To identify whether model-level or problem-level reasoning factors affect diagnostic performance, and whether the effects differ across solution categories, we analyze four factors:

Model vs. solution context: We compare variance (η^2) explained by model selection and feedback condition to determine whether diagnostic accuracy is driven by model capability or solution context.

Step complexity: We test whether step complexity predicts misdiagnosis to separate reasoning difficulty from model-level diagnostic failures. Nested logical structures are a known source of reasoning difficulty for both humans (Johnson-Laird and Wason, 1970; Shabrina et al., 2024) and LLMs (Tithi et al., 2025). Full computation details are in Appendix E.1.

Distance to goal: We test whether distance to goal (Section 3.3) predicts OR and OV at different derivation stages. Together with step complexity, this helps determine whether failures reflect problem difficulty or model-level biases.

Inference rule: We compare F1 scores across inference rules to identify rules systematically misdiagnosed across models.

Table 1: Rubric for evaluating pedagogical quality of agent-generated feedback.

Dimension	Score 1: Low	Score 2: Moderate	Score 3: High
Correctness	Contains errors that would mislead the student	Minor imprecisions but no fundamental errors	All statements are logically and factually accurate
Error Identification	Fails to identify the error or identifies incorrectly	Indicates error exists but identification is vague	Precisely identifies the specific error; affirms correctness if applicable
Revealing	Guides without revealing, using scaffolding or Socratic questioning	Hints at solution direction without explicit disclosure	Discloses solution content explicitly (names rule, states next step)
Actionability	No actionable guidance; student cannot proceed	General guidance but lacks specificity	Clear, specific guidance enabling appropriate next step

5.2 Human Evaluation (RQ3)

To determine whether better solution diagnosis leads to pedagogically appropriate feedback, two annotators independently evaluated feedback qual-

ity on 100 solution-feedback pairs per condition. We use four rubric dimensions adapted from prior work (Maurya et al., 2025), shown in Table 1. These dimensions directly operationalize the distinction between solution diagnosis and pedagogical feedback quality central to RQ3. Complete details on sample and annotator selection, annotation procedures (calibration and inter-rater reliability), and instructions are provided in Appendix I.

6 Results

We organize results around three research questions. First, we report solution classification performance and systematic errors, i.e., OR and OV *(RQ1)*. Next, we examine how model selection, feedback condition, step complexity, distance to goal, and inference rule affect classification accuracy *(RQ2)*. Finally, we assess whether accurate solution classification translates into pedagogically appropriate feedback *(RQ3)*.

6.1 RQ1: Classification Performance and Error Patterns

Models showed distinct classification patterns across student solutions. Table 2 presents classification performance (F1) across models and feedback conditions. All models achieved near-ceiling performance on optimal student solutions ($F1$: $94 - 99\%$) but struggled with valid-alternatives ($F1$: $0 - 76\%$) and incorrect solutions ($F1$: $4 - 55\%$). GPT-4.1 and GPT-o3 achieved balanced but moderate performance on valid-alternative solutions. Gemini and DeepSeek achieved the highest valid-alternative scores ($F1$: 70–76%) but poor incorrect detection ($F1$: 4-19%). In contrast, LLaMA 3 showed near-zero valid-alternative diagnosis ($F1$: 0–17%) but the highest incorrect detection ($F1$: 45-55%).

Table 2 further quantifies these patterns using conditional error probabilities, OR and OV. Gemini and DeepSeek showed high over-validation (69-71%) but low over-rejection (13-17%), treating most student solutions as valid-alternative regardless of correctness. LLaMA 3 showed the opposite pattern: 91% over-rejection but only 6% over-validation, rejecting nearly all non-optimal solutions. GPT-4.1 and GPT-o3 showed moderate rates of both errors (29-41%).

6.2 RQ2: Factors Affecting Classification

Given the substantial variation in classification performance across models, we examined the factors affecting misdiagnosis.

Table 2: Classification F1 and error rates by model. P = Peer, T = Teacher, J = Judge. OR = Over-Rejection, OV = Over-Validation. CIs: ±5–8%. $^{\dagger}N < 50$.

	Opt.	Valid-Alt.			Incorrect			OR	OV
Model	**Avg**	**P**	**T**	**J**	**P**	**T**	**J**	(%)	(%)
GPT-4.1	0.99	0.55	0.57	0.49	0.42	0.40	0.41	38.57	28.65
GPT-o3	0.99	0.48	0.55	0.45	0.38	0.40	0.40	40.50	29.97
Gemini-1.5 Pro	0.99	0.70	0.73	0.72	0.09	0.04	0.08	12.77	70.55
DeepSeek	0.99	0.74	0.76	0.74	0.09	0.12	0.19	17.12	68.60
Qwen	0.99	0.58	0.42	0.54	0.11	0.07	0.12	54.65	28.30
Mistral	0.95	0.58	0.47	0.52	0.15	0.09	0.12	26.28	54.37
Llama 3†	0.94	0.03	0.00	0.17	0.48	0.55	0.45	91.07	6.47

Table 3: Mean complexity (correct vs. incorrect steps)

Type	**Corr.**	**Incorr.**	**Diff.**	**N**
Optimal	1.95	2.46	+0.51*	1189
Valid Alt.	3.62	2.97	−0.65	816
Incorrect	3.91	3.78	−0.13	1619

$^{*}p < .05$ (Mann-Whitney U)

– ***Model Selection vs. Feedback Conditions:*** Model selection explained most of the variance in classification performance ($\eta^2 > 0.95, p < .001$), while feedback conditions had negligible effect ($\eta^2 < .01$). Table 2 shows that additional information access did not improve classification when student solutions diverged from the expected path (valid-alternative or incorrect solutions).

– ***Step Complexity:*** Because feedback conditions had negligible effect on diagnosis ($\eta^2 < .01$), we pooled feedback agents across models to maximize statistical power for complexity analysis. However, complexity predicted misdiagnosis only for Optimal solutions: more complex steps were misdiagnosed more often ($M = 2.46$) than correctly classified ones ($M = 1.95; p < .05$).

For valid-alternative and incorrect solutions, complexity did not predict errors ($p > .08$). Feedback agents rejected simpler valid-alternative solutions ($M = 2.97$) more often than complex ones ($M = 3.62$), suggesting rejection was driven by deviation from the expert path rather than step complexity. Step complexity explained negligible variance in diagnosis performance ($\eta^2 < .01, p > .05$) compared to model selection ($\eta^2 > .95, p < .001$), confirming that failures are model-level rather than difficulty-driven.

To analyze complexity effects on OR and OV, proof states were grouped into low (1-2), medium (3-4), and high (5+) complexity levels, approximately corresponding to the 33rd and 66th percentile of complexity distribution ($P_{33} = 2$, $P_{66} =$

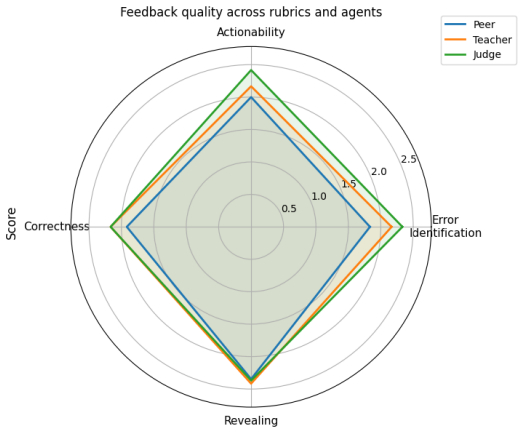


Figure 3: Mean feedback quality scores across rubric dimensions (N=100).

3). Figure 18 shows that LLaMA over-rejection approaches 100% at higher complexity tiers, while DeepSeek and Gemini over-validation persists across all tiers, further supporting model-level rather than difficulty-driven failure.

– ***Distance to Conclusion:*** Distance tiers used the same percentile-based partition: near ($d \leq 2$), mid ($2 < d \leq 3$), and far ($d > 3$). Figure 19 shows that over-rejection is highest at near distances, meaning agents most often penalize valid-alternative reasoning when solutions are closest to the conclusion. Distance to conclusion explained negligible variance ($\eta^2 < .01$), again supporting model-level rather than position-driven failures.

– ***Inference Rules:*** Analysis of inference rules and F1 scores (Figure 20) shows that failures concentrate on structurally complex rules (MT, CD, DeM). LLaMA showed minimal gains across most rule types, while Gemini and DeepSeek consistently achieved the highest scores.

6.3 RQ3: Feedback Quality

To assess whether accurate solution classification translates into pedagogically appropriate feedback, we report aggregate scores and LLM responses.

Overall patterns: Figure 3 summarizes mean rubric scores (1–3 scale). Error Identification here measures whether feedback precisely identified the reasoning flaw, distinct from classification performance (RQ1), which captures only categorical judgment. Across agents, scores were lowest on Error Identification and Actionability, indicating that feedback frequently failed to diagnose errors or provide clear guidance, consistent with automated analysis (Section 6.1 and 6.2).

Qualitative analysis further explains this pattern. Although prompts instructed agents to acknowledge correct reasoning *before addressing errors*, agents often praised solutions without identifying errors or giving actionable guidance. This produced encouraging but pedagogically misleading feedback, especially for incorrect and valid-alternative solutions. (Example 1 and 2 below)

Peer feedback: Peer agents consistently received the lowest scores across all human-evaluated dimensions (Figure 3). Limited solution context led Peer agents to restate student incorrect reasoning or provide vague feedback without diagnosing errors or offering guidance (Example 1 below and Figure 8 in Appendix H).

Example 1: *Vague Peer feedback (repeats student's mistakes)*
Proof state: $1.((S \rightarrow Y) \vee (I \wedge Q))$, $2.((I \wedge Q) \rightarrow D)$, $3.\neg D$, $4.((S \rightarrow Y) \rightarrow D)$, $5.\neg(S \rightarrow Y)$, $6.(I \wedge Q)$
Solution context: D
Student reasoning: *Use Modus Tollens on* $((S \rightarrow Y) \rightarrow D)$ *and* $\neg D$*...*
Student error: Rule cant be applied on parent statements
Peer Feedback: *"Great job on identifying Modus Tollens on* $((S \rightarrow Y) \rightarrow D)$ *and* $\neg D$ *as a key step. Now ask yourself, is there a step that could bridge your current progress to the CONCLUSION more efficiently..."*

Teacher and Judge feedback: Teacher and Judge agents performed better at error identification than Peer agents. However, their feedback often penalized valid-alternative reasoning by redirecting students toward the provided solution context. This pattern was strongest for complex inference rules, which were also associated with higher error rates in Section 6.2 (Figure 20). In contrast, agents more readily acknowledged valid-alternative reasoning for simpler inference rules (Example 2, 3 and Figure 9).

Example 2: Teacher feedback validating incorrect reasoning
Proof state: $1.(B \leftrightarrow \neg J)$, $2.(\neg N \vee J)$, $3.(B \vee \neg N)$, $4.((B \rightarrow \neg J) \wedge (\neg J \rightarrow B))$, $5.(N \rightarrow J)$, $6.(\neg B \rightarrow \neg N)$
Solution context: *Derive* $(J \vee \neg N)$ *from* $(\neg N \vee J)$ *using Commutative rule.*
Student reasoning: $(B \rightarrow \neg N)$ *is optimal because it is the exact form of the conclusion, it can be derived from the contrapositive of* $(\neg B \rightarrow \neg N)$*...*
Student error: Incorrect derivation
Teacher Feedback: *"Great job recognizing that* $(B \rightarrow \neg N)$ *is the contrapositive of* $(\neg B \rightarrow \neg N)$ *and using that to reach the conclusion..."*

Verification Condition Contrary to prior LLM-as-Judge findings (Zheng et al., 2023), Judge feedback did not consistently improve feedback quality. When Peer agents missed errors, Judge agents often propagated rather than corrected them (Figure 12, anchoring Judge feedback to the Peer's framing inheriting the or exacerbating mistakes. This

asymmetric pattern, where verification does not consistently improves some response is consistent with our prior work (Yasir et al., 2026).

Example 3: *Judge repeats Peers' mistakes and forces towards the available solution context*
Proof state: $1.\neg(\neg G \wedge B), \quad 2.(G \rightarrow D)$
Student reasoning: *Applying DN to* $\neg(\neg G \wedge B))$ *to get* $(G \wedge B)$ *is optimal because it directly gives us both G and B as available conjuncts...*
Student error: Incorrect rule application
Peer Feedback: *"Great job applying double negation here! As you move forward, consider how you can use the new conjuncts to get closer..."*
Solution context: *Derive* $(\neg\neg G \vee \neg B)$ *from* $\neg(\neg G \wedge B)$ *using DeM Law.*
Judge Feedback: *"You made a good attempt by considering double negation, but before applying DN, try using De Morgan's Law to rewrite..."*

7 Discussion

RQ1: Consistent with the high OR and OV rates reported in Section 6.1, LLMs showed limited sensitivity to valid-alternative reasoning, relying on surface alignment with the provided solution context rather than reasoning validity. These failures have direct pedagogical consequences: OR discourages productive exploration and penalizes sound reasoning that diverges from the expert path (Maniktala et al., 2023), while OV reinforces misconceptions and limits corrective feedback essential for learning (VanLehn, 2006). These failure modes are obscured under binary correctness schemes and only separable through our three-way KG-grounded framework, highlighting the importance of evaluation methodology alongside model capability (VanLehn, 2006; Koedinger et al., 2012).

RQ2: As shown in Section 6.2, model selection explained substantially more variance than feedback condition, indicating that diagnostic failures are primarily model-level rather than informational. Models performed well when solutions aligned with the optimal path but showed systematic bias once reasoning diverged, regardless of the provided solution context. Together with the negligible effects of step complexity, distance to conclusion, and inference rule type, these findings suggest persistent limits in LLM reasoning for constrained domains such as logic proofs. Improving diagnosis therefore likely requires capability-level advances rather than prompt or information-access design (Stamper et al., 2024; Venugopalan et al., 2025).

RQ3: Section 6.3 further showed that accurate diagnosis does not reliably translate into pedagogically effective feedback. Agents with more solution context produced overly directive feedback, while less-informed agents generated open-ended responses lacking diagnostic grounding. This reflects the assistance dilemma: excessive directness constrains learning, while insufficient guidance fails to support progress (Koedinger and Aleven, 2007). Prompt-level layering alone therefore cannot ensure adaptive feedback. Effective verification likely requires independent diagnostic signals, similar to the grounded student modeling mechanisms used in ITS. Our findings further suggest that verification may require distinct models or explicit reasoning chains enforcing independent evaluation rather than sequential refinement of flawed judgments.

LLMs address ITS limitations in conversational flexibility and cross-domain scalability (Reddig et al., 2025); however, their inability to reliably distinguish valid-alternative from incorrect reasoning risks pedagogically misleading feedback, precisely the problem ITS student modeling is designed to address (Venugopalan et al., 2025). These findings confirm that linguistic fluency alone does not ensure pedagogical effectiveness (Shetye, 2024). Rather than replacing ITS, a more effective hybrid architecture may use LLMs for open-ended dialogue and hint generation while KG-grounded classifiers provide diagnostic labeling, with feedback conditioned on classifier outputs instead of self-assessed reasoning validity.

8 Conclusion

This study benchmarked LLMs as step-level tutors in propositional logic, evaluating both diagnostic accuracy and feedback quality across conditions that vary in solution information access. Our results show that LLMs reliably confirmed optimal steps but struggled with valid-alternative and incorrect solutions, precisely where adaptive tutoring is most critical. Classification failures on valid-alternative and incorrect solutions persisted across all models regardless of solution context, with model selection governing the magnitude of errors. Moreover, accurate classification did not reliably translate into pedagogically actionable feedback. These findings indicate that current LLMs cannot resolve the assistance dilemma without ITS-grounded diagnostic mechanisms. Effective LLM-based tutoring in structured reasoning domains therefore requires hybrid architectures that delegate diagnostic classification to KG-grounded models while leveraging LLMs for open-ended scaffolding and dialogue.

Limitations

This study targets propositional logic by design: KG-grounded exhaustive enumeration of valid inference paths is feasible in this formal domain and provides the principled ground truth that evaluation requires. Findings may not directly generalize to domains where exhaustive solution space enumeration is not tractable or where symbolic constraints are weaker.

Our 516 proof states are drawn from a single undergraduate Discrete Mathematics course at one institution across five difficulty levels, which limits the diversity of reasoning contexts and problem types represented.

Additionally, while LLM-simulated solutions enable controlled evaluation across all three solution categories, they may not fully capture the diversity of authentic student reasoning; planned extensions will investigate alignment between simulated and real student solution patterns using interaction logs.

Feedback agents were evaluated using zero-shot prompting after iterative prompt refinement and we did not see significant performance difference with few-shot prompting while the initial prompt calibration. However, we expect fine-tuning may improve classification performance, particularly for valid-alternative and incorrect categories, and remains a direction for future work.

This study diagnoses systematic classification failures without proposing a mitigation mechanism. The empirical findings are intended to inform the design of hybrid architectures that combine LLM flexibility with KG-grounded diagnostic precision; implementing and evaluating such systems is left to future work.

Our 10,836 solution-feedback pairs capture single-step feedback evaluation and do not model multi-turn student-tutor interactions or cumulative learning gains over time, which are ultimately what adaptive tutoring aims to improve.

Finally, our human evaluation covers 100 solution-feedback pairs per condition across four rubric dimensions; systematic fine-grained analysis of cases where models correctly classify a solution yet still produce inaccurate, vague, revealing, or unactionable feedback would require annotation at a scale, and represents an open methodological challenge for the field.

Ethics Statement

Student interaction data was collected under IRB approval with informed consent and anonymized before analysis. Two annotators with domain expertise in the tutoring system served as expert raters; both are co-authors of this study. To mitigate potential conflict of interest, annotators coded independently with no access to each other's labels, research hypotheses, or study outcomes during annotation, and rubric definitions were finalized before coding began.

Reproducibility

Code and data can be found here [2]. All experiments use temperature 0.0 to ensure deterministic, reproducible results, prioritizing controlled comparison over response diversity. While real-world tutoring systems might benefit from non-zero temperature, our evaluation focuses on measuring the systematic effects of multi-agent architectures under controlled conditions. Model specifications and prompts appear in Appendices G and J.

AI Usage Disclosure

This work studies and evaluates large language models as research objects. We utilized large language models as assistive tools during manuscript preparation, including formatting guidelines and brainstorming organization, mainly, and paraphrasing on a need basis only. We did not use any AI tools for designing, implementing, or executing this research study. All the claims, analyses, hypotheses, and conclusions are developed, verified, and reviewed by the authors. Moreover, no AI tool was used for generating or labeling data, making judgments about data, or making any scientific claims. All implementations were reviewed, validated, and finalized by the authors. The authors take full responsibility for the correctness, originality, and integrity of the work.

Acknowledgment

This research was partially supported by the NSF Grant: Generalizing Data-Driven Technologies to Improve Individualized STEM Instruction by Intelligent Tutors (2013502).

[2] https://github.com/tahreemm/BEA_2026

References

Vincent Aleven, Bruce Mclaren, Jonathan Sewall, and Kenneth Koedinger. 2009. A New Paradigm for Intelligent Tutoring Systems: Example-Tracing Tutors. *I. J. Artificial Intelligence in Education*, 19:105–154.

Tiffany Barnes and John Stamper. 2008. Toward Automatic Hint Generation for Logic Proof Tutoring Using Historical Student Data. pages 373–382.

Luca Benedetto, Giovanni Aradelli, Antonia Donvito, Alberto Lucchetti, Andrea Cappelli, and Paula Buttery. 2024. Using LLMs to simulate students' responses to exam questions. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 11351–11368, Miami, Florida, USA. Association for Computational Linguistics.

Conrad Borchers and Tianze Shou. 2025. Can large language models match tutoring system adaptivity? a benchmarking study. In *International Conference on Artificial Intelligence in Education*, pages 407–420. Springer.

Liuqiao Chen, Yan Huang, Zhenglin Jiang, and Ying Chen. 2025. Generative Feedback for Code Learning: A Study on LLM-Driven Formative Assessment. In *Proceedings of the 2025 International Conference on AI-enabled Education*, AIEE '25, pages 253–259, New York, NY, USA. Association for Computing Machinery.

Cornelia Große and Alexander Renkl. 2006. Effects of multiple solution methods mathematics learning. Learning and Instruction, 16(2), 122-138. *Learning and Instruction - LEARN INSTR*, 16:122–138.

Shuchen Guo, Ehsan Latif, Yifan Zhou, Xuan Huang, and Xiaoming Zhai. 2024. Using Generative AI and Multi-Agents to Provide Automatic Feedback. *arXiv preprint*. ArXiv:2411.07407 [cs].

Adit Gupta, Jennifer Reddig, Tommaso Calo, Daniel Weitekamp, and Christopher J MacLellan. 2025. Beyond final answers: Evaluating large language models for math tutoring. In *International Conference on Artificial Intelligence in Education*, pages 323–337. Springer.

Simeng Han, Hailey Schoelkopf, Yilun Zhao, Zhenting Qi, Martin Riddell, Wenfei Zhou, James Coady, David Peng, Yujie Qiao, Luke Benson, Lucy Sun, Alex Wardle-Solano, Hannah Szabo, Ekaterina Zubova, Matthew Burtell, Jonathan Fan, Yixin Liu, Brian Wong, Malcolm Sailor, and 16 others. 2024. FOLIO: Natural Language Reasoning with First-Order Logic. *arXiv preprint*. ArXiv:2209.00840 [cs].

Amogh Inamdar, Uzay Macar, Michel Vazirani, Michael Tarnow, Zarina Mustapha, Natalia Dittren, Sam Sadeh, Nakul Verma, and Ansaf Salleb-Aouissi. 2025. Logiclearner: A tool for the guided practice of propositional logic proofs. *arXiv preprint arXiv:2503.19280*.

Qinjin Jia, Jialin Cui, Ruijie Xi, Chengyuan Liu, Parvez Rashid, Ruochi Li, and Edward Gehringer. 2024. On assessing the faithfulness of llm-generated feedback on student assignments. In *Proceedings of the 17th International Conference on Educational Data Mining*, pages 491–499.

PN Johnson-Laird and PC Wason. 1970. Insight into a logical relation. *The Quarterly Journal of Experimental Psychology*, 22(1):49–61.

Kenneth R Koedinger and Vincent Aleven. 2007. Exploring the assistance dilemma in experiments with cognitive tutors. *Educational psychology review*, 19(3):239–264.

Kenneth R Koedinger, Albert T Corbett, and Charles Perfetti. 2012. The knowledge-learning-instruction framework: Bridging the science-practice chasm to enhance robust student learning. *Cognitive science*, 36(5):757–798.

Rongxin Liu, Julianna Zhao, Benjamin Xu, Christopher Perez, Yuliia Zhukovets, and David Malan. 2025a. Improving AI in CS50: Leveraging Human Feedback for Better Learning. pages 715–721.

Rongxin Liu, Julianna Zhao, Benjamin Xu, Christopher Perez, Yuliia Zhukovets, and David J Malan. 2025b. Improving ai in cs50: Leveraging human feedback for better learning. In *Proceedings of the 56th ACM Technical Symposium on Computer Science Education V. 1*, pages 715–721.

Xinyi Lu and Xu Wang. 2024. Generative students: Using LLM-simulated student profiles to support question item evaluation. In *Proceedings of the Eleventh ACM Conference on Learning @ Scale (L@S '24)*, pages 16–27, Atlanta, GA, USA. ACM.

Jakub Macina, Nico Daheim, Sankalan Chowdhury, Tanmay Sinha, Manu Kapur, Iryna Gurevych, and Mrinmaya Sachan. 2023. MathDial: A Dialogue Tutoring Dataset with Rich Pedagogical Properties Grounded in Math Reasoning Problems. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 5602–5621, Singapore. Association for Computational Linguistics.

Hamed Mahdavi, Pouria Mahdavinia, Samira Malek, Pegah Mohammadipour, Alireza Hashemi, Majid Daliri, Alireza Farhadi, Amir Khasahmadi, Niloofar Mireshghallah, and Vasant Honavar. 2025. RefGrader: Automated Grading of Mathematical Competition Proofs using Agentic Workflows. *arXiv preprint*. ArXiv:2510.09021 [cs].

Mehak Maniktala, Min Chi, and Tiffany Barnes. 2023. Enhancing a student productivity model for adaptive problem-solving assistance. *User Modeling and User-Adapted Interaction*, 33(1):159–188.

Amogh Mannekote, Adam Davies, Jina Kang, and Kristy Elizabeth Boyer. 2025. Can llms reliably simulate human learner actions? a simulation authoring framework for open-ended learning environments. In

Proceedings of the AAAI Conference on Artificial Intelligence, volume 39, pages 29044–29052.

Kaushal Kumar Maurya, Kv Aditya Srivatsa, Kseniia Petukhova, and Ekaterina Kochmar. 2025. Unifying AI Tutor Evaluation: An Evaluation Taxonomy for Pedagogical Ability Assessment of LLM-Powered AI Tutors. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 1234–1251, Albuquerque, New Mexico. Association for Computational Linguistics.

Tung Phung, Victor-Alexandru Pădurean, Anjali Singh, Christopher Brooks, José Cambronero, Sumit Gulwani, Adish Singla, and Gustavo Soares. 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*, LAK '24, pages 12–23, New York, NY, USA. Association for Computing Machinery.

Jennifer M. Reddig, Arav Arora, and Christopher J. MacLellan. 2025. Generating In-Context, Personalized Feedback for Intelligent Tutors with Large Language Models. *International Journal of Artificial Intelligence in Education*, 35(6):3459–3500.

Bárbara Rodrigues, Rui Pinto, and Gil Gonçalves. 2025. A systematic literature review of ai-driven intelligent tutoring systems in engineering education: Emphasizing personalization, feedback, and student monitoring. *IEEE Access*, 13:190152–190177.

Abulhair Saparov and He He. 2023. Language Models Are Greedy Reasoners: A Systematic Formal Analysis of Chain-of-Thought. *arXiv preprint*. ArXiv:2210.01240 [cs].

Iulian Vlad Serban and 1 others. 2020. Automated personalized feedback improves learning gains in an intelligent tutoring system. In *International Conference on Artificial Intelligence in Education*, pages 140–146. Springer. Personalized hints improve learning gains; feedback from ZPD.

Preya Shabrina, Behrooz Mostafavi, Mark Abdelshiheed, Min Chi, and Tiffany Barnes. 2024. Investigating the impact of backward strategy learning in a logic tutor: Aiding subgoal learning towards improved problem solving. *International Journal of Artificial Intelligence in Education*, 34(3):825–861.

Shamini Shetye. 2024. An evaluation of khanmigo, a generative ai tool, as a computer-assisted language learning app. *Studies in Applied Linguistics and TESOL*, 24(1).

Yao Shi, Rongkeng Liang, and Yong Xu. 2025. Educationq: Evaluating llms' teaching capabilities through multi-agent dialogue framework. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 32799–32828.

John Stamper, Ruiwei Xiao, and Xinying Hou. 2024. Enhancing llm-based feedback: Insights from intelligent tutoring systems and the learning sciences. In *International Conference on Artificial Intelligence in Education*, pages 32–43. Springer.

Oyvind Tafjord, Bhavana Dalvi Mishra, and Peter Clark. 2020. Proofwriter: Generating implications, proofs, and abductive statements over natural language. *arXiv preprint arXiv:2012.13048*.

Sutapa Dey Tithi, Arun Kumar Ramesh, Clara DiMarco, Xiaoyi Tian, Nazia Alam, Kimia Fazeli, and Tiffany Barnes. 2025. The promise and limits of LLMs in constructing proofs and hints for logic problems in intelligent tutoring systems. *arXiv preprint*. ArXiv:2505.04736 [cs].

Kurt VanLehn. 2006. The behavior of tutoring systems. *International journal of artificial intelligence in education*, 16(3):227–265.

Kurt VanLehn. 2011. The relative effectiveness of human tutoring, intelligent tutoring systems, and other tutoring systems. *Educational Psychologist*, 46(4):197–221. Step-based ITS achieve effect sizes of 0.75-0.80, comparable to human tutoring (0.79).

Devika Venugopalan, Ziwen Yan, Conrad Borchers, Jionghao Lin, and Vincent Aleven. 2025. Combining large language models with tutoring system intelligence: A case study in caregiver homework support. In *Proceedings of the 15th International Learning Analytics and Knowledge Conference*, pages 373–383.

Daniel Weitekamp, Erik Harpstead, and Ken R. Koedinger. 2020. An Interaction Design for Machine Teaching to Develop AI Tutors. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, CHI '20, pages 1–11, New York, NY, USA. Association for Computing Machinery.

Tahreem Yasir, Sutapa Dey Tithi, Benyamin Tabarsi, Dmitri Droujkov, Sam Gilson Yasitha Rajapaksha, Xiaoyi Tian, Arun Ramesh, Tiffany Barnes, and 1 others. 2026. When verification hurts: Asymmetric effects of multi-agent feedback in logic proof tutoring. *arXiv preprint arXiv:2603.27076*.

Mohamed Zerkouk and Belkacem Chikhaoui. 2025. A comprehensive review of ai-based intelligent tutoring systems: Applications and challenges. *arXiv preprint arXiv:2507.18882*. ITS can improve student performance; CLT informs stepwise hint design.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, and 1 others. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in neural information processing systems*, 36:46595–46623.

A Inference Rule List

We employ a fixed set of propositional inference rules used in Logic Tutor for the dataset. The short names were used for consistent response generation and evaluation. The complete list of inference rules, along with short names and derivations are provided below in Table 4.

Table 4: Propositional inference rules used in this work.

Abbrev.	Rule Name	Form
MP	Modus Ponens	$P \to Q,\ P \Rightarrow Q$
MT	Modus Tollens	$P \to Q,\ \neg Q \Rightarrow \neg P$
Conj	Conjunction	$P,\ Q \Rightarrow P \wedge Q$
Simp	Simplification	$P \wedge Q \Rightarrow P$ (or Q)
Add	Addition	$P \Rightarrow P \vee Q$
DS	Disjunctive Syllogism	$P \vee Q,\ \neg P \Rightarrow Q$
HS	Hypothetical Syllogism	$P \to Q,\ Q \to R$ $\Rightarrow P \to R$
Impl	Implication	$P \to Q \equiv \neg P \vee Q$
DN	Double Negation	$P \equiv \neg\neg P$
CP	Contraposition	$P \to Q \equiv \neg Q \to \neg P$
Com	Commutation	$P \vee Q \equiv Q \vee P$
Assoc	Associativity	$(P \vee Q) \vee R$ $\equiv P \vee (Q \vee R)$
Dist	Distribution	$P \wedge (Q \vee R)$ $\equiv (P \wedge Q) \vee (P \wedge R)$
CD	Constructive Dilemma	$(P \to Q), (R \to S), P \vee R$ $\Rightarrow Q \vee S$
Equiv	Equivalence	$P \leftrightarrow Q$ $\equiv (P \to Q) \wedge (Q \to P)$

B Dataset Distribution

Table 5: Dataset distribution across practice levels (2-6)

Level	Proof States	Avg. Statements
2 (Introductory)	48	5.86
3 (Basic)	111	7.73
4 (Intermediate)	148	7.94
5 (Advanced)	85	5.64
6 (Expert)	95	6.41
Total	**516**	**6.72**

*Levels 1 and 7 excluded because of unavailability of hints.

C Illustrative Logic Tutor Proof Interaction

Figure 4–6 illustrates a representative student interaction in the propositional logic tutor. These screenshots demonstrate forward chaining, rule application, and goal completion within the tutor interface.

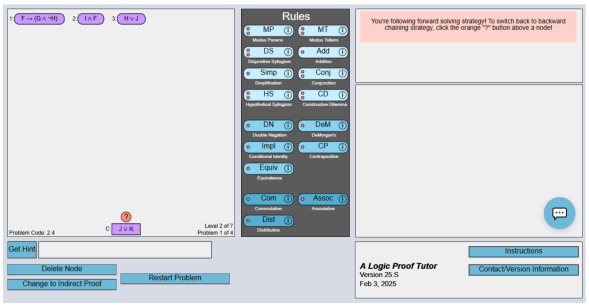


Figure 4: **Initial proof state and goal specification.** The student is presented with the premises (top left) and the target conclusion ($J \vee K$) at the bottom. Available inference rules are displayed on the middle. At this stage, no intermediate steps have been derived, and the student must choose a productive forward step toward the goal.

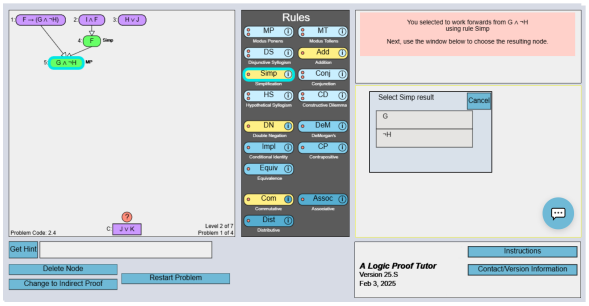


Figure 5: **Rule application with guided simplification.** After deriving an intermediate conjunction via Modus Ponens, the student applies the *Simplification* rule. The interface prompts the learner to select the appropriate resulting literal (G or $\neg H$), illustrating fine-grained, step-level decision making supported by rule constraints.

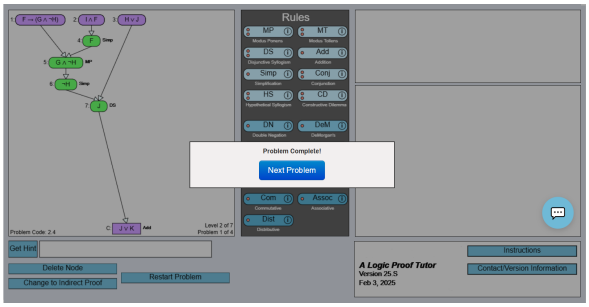


Figure 6: **Successful proof completion.** The student derives J via Disjunctive Syllogism and applies the *Addition* rule to reach the target conclusion $J \vee K$. The system confirms completion, reinforcing correct rule sequencing and alignment with the goal state.

D Representative Proof State

This appendix presents an illustrative propositional logic proof instance Figure 7 annotated with inter-

Representative proof Instance

Givens

- (1) $((S \to Y) \lor (I * Q))$
- (2) $((I \land Q) \to D)$
- (3) $\neg D$
- (4) $((S \to Y) \to D)$

Intermediate Steps

- (5) $\neg(S \to Y)$ [MT: (3), (4)]
- (6) $\neg(I \land Q)$ [MT: (3), (2)]
- (7) $(S \to Y)$ [DS: (6), (1)]

Conclusion: Y

Peer Solution Context

$\neg(\neg S \lor Y)$

Teacher & Judge Solution Context

Derive $\neg(\neg S \lor Y)$ from $\neg(S \to Y)$ using the Implication rule.

Figure 7: Illustrative proof instance showing intermediate derivations and differential hint access for Peer and Teacher agents.

mediate derivations and agent-specific hints. Intermediate steps are derived by applying valid inference rules to previously established statements, with each step indexed and linked to its parent statements and rule application. This explicit structure exposes the shortest derivation path toward the conclusion while preserving alternative valid reasoning trajectories.

The Peer is provided only with the KG-verified optimal next symbolic step, enabling guidance that focuses on local progression without access to derivational context. In contrast, the Teacher additionally receives the rule and parent statements used to derive this step, allowing feedback to reference structural reasoning without revealing the answer directly. This distinction isolates the effect of solution access: the Peer operates under minimal grounding, while the Teacher leverages full derivational context to support more informed intervention. The representative proof state is available in Figure 7.

E Knowledge Graph based Evaluation Metrics

This appendix provides formal definitions and illustrative examples of the graph-based metrics used to evaluate next-step reasoning quality. All metrics are computed with respect to the knowledge graph (KG) constructed for each problem.

E.1 Step Complexity

Step complexity is computed as a nesting-weighted operator count:

$$c(\phi) = \sum_{o \in \mathcal{O}(\phi)} w(o, \phi)$$

where $w(o, \phi) = w_{\text{base}}(o) + w_{\text{nest}}(o)$ if o applies to a parenthesized subexpression in ϕ, and $w(o, \phi) = w_{\text{base}}(o)$ otherwise. Each opening parenthesis contributes an additional unit to capture structural depth.

Higher-order connectives receive larger nesting penalties because applying them to compound arguments requires tracking negation across multiple sub-propositions simultaneously (Johnson-Laird and Wason, 1970; Tithi et al., 2025). Atomic expressions containing no operators receive a score of 0.

For example: $c(F) = 0$; $c(A \lor B) = 1$; $c(A \to (B \lor C)) = 6$, where the implication receives its base weight plus a nesting penalty for applying to the parenthesized subexpression $(B \lor C)$.

F Implementation Details

Response Constraints. We standardize inference rule references using abbreviations (MP, MT, DS, HS, DeM, Impl) for consistent evaluation. The complete list appears in Table 4.

Token Limits. Rather than hard limits, we employ soft constraints via prompts (2–3 sentences).

Quality Validation. Two automated mechanisms ensure data integrity: (1) zero-token outputs trigger automatic retry, and (2) responses missing required JSON fields or violating constraints trigger re-prompting. Persistent failures after three attempts are flagged for manual review. Across 516 instances on 7 models, fewer than 3% required retry.

G Model Specifications

Please refer to Table 6.

Table 6: Models used in our experiments

Model	Org.	API	Context	Param
GPT-4.1	OpenAI	OpenAI	1M	-
GPT-o3	OpenAI	OpenAI	1M	-
Gemini 1.5 Pro	Google	Vertex	128K	-
Qwen-3-32B	Alibaba	Groq	32K	32B
LLaMA-3.3-70B	Meta	Groq	128K	70B
Mistral Large	Mistral	Mistral	256K	675B
Deepseek-r1-0528	Deepseek	Deepinfra	128K	671B

"-" refers to undisclosed

H Representative Examples of Failure Modes

Please refer to Figures 8, 9, and 12.

Proof state: $1.((S \to Y) \vee (I \wedge Q))$, $2.((I \wedge Q) \to D)$, $3.\neg D$, $4.((S \to Y) \to D)$, $5.\neg(S \to Y)$, $6.(I \wedge Q)$
Reasoning: *Use Modus Tollens on* $((S \to Y) \to D)$ *and* $\neg D$*...*
Reasoning error: Incorrect rule used
Solution context : D
Peer Feedback: *"Great job on identifying Modus Tollens on* $((S \to Y) \to D)$ *and* $\neg D$ *as a key step. Now ask yourself, is there a step that could bridge your current progress to the CONCLUSION more efficiently..."*

Figure 8: Example for **Vague Peer Feedback** - *repeats student's incorrect solution*

Proof state: $1.(B \leftrightarrow \neg J)$, $2.(\neg N \vee J)$, $3.(B \vee \neg N)$, $4.((B \to \neg J) \wedge (\neg J \to B))$, $5.(N \to J)$, $6.(\neg B \to \neg N)$
Student Reasoning: $(B \to \neg N)$ *is optimal because it is the exact form of the conclusion, it can be derived from the contrapositive of* $(\neg B \to \neg N)$*...*
Reasoning error: Incorrect derivation
Solution context:*Derive* $(J \vee \neg N)$ *from* $(\neg N \vee J)$ *using Commutative rule.*
Teacher Feedback: *"Great job recognizing that* $(B \to \neg N)$ *is the contrapositive of* $(\neg B \to \neg N)$ *and using that to reach the conclusion..."*

Figure 9: Example for **Teacher feedback validating incorrect reasoning**

Proof state:
$(S \to D) \vee I, (-S \vee Q) \to Y, \neg D, \neg D \to \neg I$
Student Reasoning: MT on $\neg D > \neg I$ and $\neg D$ is needed to get $\neg I$ to be used later with $(S > D) \vee I$ for DS
Solution Context: Derive $\neg S$ from $(S \to D)$ and $\neg D$ using *Modus Tollens (MT)*.
Teacher Feedback: Great job identifying the relevant candidates! You correctly spotted that $(\neg D \to \neg I)$ and $\neg D$ could be useful, but there's a small mix-up with the rule. Recall: MT requires a negated *consequent*, while MP works with a *true antecedent*. How you might use MP on $(S \to D)$ and $\neg D$ to get $\neg D$.

Figure 10: Teacher (complete hint) feedback reveals solution on **incorrect** student response

Proof State (Excerpt)
$(S \to D), ((\neg S \vee Q) \to Y), \neg D, (\neg D \to \neg I), ...$
Student Reasoning: $\neg S$ *is optimal because we already have* $(S \to D)$ *from INTERMEDIATE STEPS, and simplifying it to* $\neg S$ *allows us to construct* $(\neg S \vee Q)$*...*
Peer Solution context: $\neg S$
Peer Feedback: *Excellent reasoning! You correctly identified how simplifying* $(S \to D)$ *helps set up the next step toward the conclusion...*
Judge Feedback: *Great job recognizing that you need -S to proceed!, think about what rule you can use on* $(S \to D)$ *to turn it into a form more suitable for Simp...*

Figure 11: Judge Verification Response, extending Incorrect Peer feedback (anchoring on incorrect student reasoning)

Proof state: $1.\neg(\neg G \wedge B), \quad 2.(G \rightarrow D)$
Solution context:*Derive* $(\neg\neg G \vee \neg B)$ *from* $\neg(\neg G \wedge B)$ *using DeM Law.*
Student reasoning: *Applying DN to* $\neg(\neg G \wedge B))$ *to get* $(G \wedge B)$ *is optimal because it directly gives us both G and B as available conjuncts...*
Reasoning error: Incorrect rule application
Peer Feedback: *"Great job applying double negation here! As you move forward, consider how you can use the new conjuncts to get closer..."*
Judge Feedback: *"You made a good attempt by considering double negation, but before applying DN, try using De Morgan's Law to rewrite..."*

Figure 12: Example for **Judge Feedback repeating Peers' mistakes** - *also directs towards the hint without recognizing the reasoning error*

H.1 Student Agent Response

We show a representative Student agent response to demonstrate candidate generation, explicit reasoning, and next-step generation. The example corresponds to a single proof instance and highlights how intermediate reasoning is externalized for downstream tutoring and verification. The sample response is shown in Figure 13.

Givens:
$((\neg K \vee L) \rightarrow (M \wedge N)), (K \rightarrow O), \neg O\ \neg K, (\neg K \vee L), (M \wedge N)$ **Conclusion:** N
Student Reasoning: Applying a rule directly to $(M \wedge N)$ is the most efficient path to the conclusion. Since $(M \wedge N)$ is available, simplification yields the target.
Next Step: N
Rule: Simp
Parent Statement: $(M \wedge N)$

Figure 13: Sample problem instance and corresponding **correct** student response

I Manual Evaluation Protocol

To complement our automated metrics and better understand qualitative differences in feedback behavior across pipelines, we conducted a manual evaluation of a representative subset of model responses. This analysis focuses on pedagogical properties of feedback that are difficult to capture through accuracy-based metrics alone, such as scaffolding quality and answer revelation.

I.0.1 Annotation Procedures

The samples were drawn from 200 distinct proof states stratified across solution categories and difficulty levels. Annotators included a graduate researcher and an instructor with expertise in the logic tutoring system, ensuring familiarity with the proof domain and pedagogical aspects of feedback. Annotators were provided with the problem context (givens, intermediates, and conclusion), the student's response, and the corresponding feedback generated by each feedback condition. Annotators worked independently and were blinded to model identity and quantitative performance results or feedback condition that generated the feedback. They were instructed to rate each feedback response independently on the four rubric dimensions based solely on the proof state, simulated solution, and generated feedback.

Annotators were trained on the rubric dimensions through an initial calibration phase using a shared sample of 20 pairs. Rubric definitions were iteratively refined, and disagreements were resolved through discussion until reaching a common interpretation of each dimension.

Due to the cognitive complexity of step-level logic feedback, during the initial calibration phase annotators agreed on a coarse 3-point ordinal scale (1–3) for all rubrics, with 1 = poor, 2 = partial, 3 = strong. This scale captures meaningful distinctions while maintaining annotation reliability and consistency. They then independently coded an additional sample until inter-rater reliability exceeded Cohen's $\kappa > 0.80$ per dimension and overall. Annotators then independently coded all remaining samples with no access to each other's labels or the research hypotheses.

I.1 Annotation Instructions

For each feedback response, rate independently on four dimensions: Correctness, Error Identification, Revealing, and Actionability. Each dimension is scored 1–3 as defined in the rubric. Base your rating solely on the proof state, the simulated student solution, and the generated feedback. For Correctness, assess whether the feedback contains logical errors that would mislead the student. For Error Identification, assess whether the feedback

precisely identifies the reasoning flaw or correctly affirms sound reasoning. For Revealing, assess the degree to which feedback discloses solution content explicitly. For Actionability, assess whether the feedback provides guidance specific enough for the student to proceed. When in doubt, refer to the calibration examples discussed during the calibration phase.

J Prompt Design

This section provides the complete prompts used for each agent in our multi-agent tutoring system. All prompts use structured JSON response formats to ensure consistent parsing.

Our prompt design translates established pedagogical principles into operational constraints for LLM agents. Each agent's prompt encodes specific instructional strategies grounded in learning science research, ensuring that generated feedback aligns with effective tutoring practices. We describe the key design decisions below.

J.1 Student Prompt

The base system message includes the Role and Task information along with instructions, whereas the user message includes the problem instance itself. The prompt template is provided below in Figure 14.

J.2 Peer Prompt

The Peer operates under *minimal hint access*: it receives only the correct next step without knowledge of the complete solution path. The prompt enforces a critical **planning-before-feedback** requirement: the Peer must first generate an internal derivation plan explaining how the correct step is derived (identifying the rule and parent statements) before producing student-facing feedback. This design decision ensures the Peer develops a genuine understanding of the solution rather than pattern-matching, aligning with research showing that scaffolded feedback improves instructional quality (Serban et al., 2020). The prompt is illustrated in Figure 15.

J.3 Teacher Prompt

The Teacher agent has access to the complete next step and provides scaffolded feedback without revealing the answer. The prompt is illustrated in Figure 16.

J.4 Judge Prompt

The Judge agent is tasked with thoroughly evaluating both the student's and the teacher's responses to the logic proof problem. While the Teacher has access to the correct step and is therefore not asked to verify its own guidance, the Judge is specifically instructed to examine and identify any errors or inaccuracies present in both the student's and the teacher's responses.

Equipped with comprehensive access to the correct derivation steps from the knowledge base, the Judge carefully analyzes the reasoning behind both responses. Its role is to provide targeted evaluations and, where necessary, concise and actionable feedback to ensure alignment with the correct logic step. This approach ensures a robust layer of verification and strengthens the overall quality and pedagogical value of the feedback offered to the student. The prompt is illustrated in Figure 17.

Student Simulation Prompt

Role: You are a Student in an undergraduate Discrete Structures course solving a propositional logic proof. Your task is to produce the *single most optimal next step* that advances the proof toward the conclusion.

Task:

1. Review the givens and intermediate steps.
2. Propose 2–3 candidate next steps.
3. Select the candidate that most directly advances toward the conclusion.
4. Justify your choice and output the selected next step.

Constraints:

- Output exactly one next step in *symbolic notation only*.
- Use only predefined inference rules (e.g., MP, MT, Conj, DS).
- Parent statements must be actual expressions, not line numbers.

Response Format:

- `CANDIDATES`: 2–3 candidate steps with brief justification
- `REASONING`: Why the selected step is optimal
- `NEXT_STEP`: Symbolic expression
- `RULE`: Inference rule (short name)
- `PARENT_STATEMENTS`: Supporting expressions

Figure 14: Base System Prompt for Student

Peer Prompt

Role: You are a Peer evaluating a student's proposed next step in a propositional logic proof, with access to the KG-derived optimal step.
Task:

1. Analyze how the optimal step is derived (rule and parent statements).
2. Evaluate the student's candidates, reasoning, and chosen next step.
3. Classify the student's step as *Correct*, *Valid Alternative*, or *Incorrect*.
4. Provide brief, scaffolded feedback guiding the student toward the optimal step.

Constraints:

- Do not reveal the optimal step, its rule, or parent statements.
- Acknowledge what the student did correctly before addressing errors.
- Use Socratic questions to guide reasoning; keep feedback concise (2–3 sentences).
- Use predefined inference rule short names only.

Response Format:

- `STUDENT_ERRORS`: Brief explanation or `Correct`
- `NEXT_STEP_CORRECTNESS`: `Correct` / `Suboptimal` / `Incorrect`
- `PEER_FEEDBACK`: Scaffolded guidance without answer revelation

Figure 15: Base System prompt for Peer

Teacher Prompt

Role: You are a Teacher evaluating a student's proposed next step in a propositional logic proof, with access to the complete solution (`KNOWLEDGE_BASE_STEPS`).

Task:

1. Compare the student's response against the knowledge-base solution.
2. Identify errors in the student's logic, rule usage, or reasoning.
3. Classify the student's next step as *Correct*, *Valid Alternative*, or *Incorrect*.
4. Provide brief, scaffolded feedback guiding the student toward the correct solution.

Constraints:

- Do not reveal the exact next step, rule, or parent statements from the solution.
- Acknowledge correct aspects of the student's attempt before addressing errors.
- Use Socratic questions to guide reasoning; keep feedback concise (2–3 sentences).
- Refer to the student's candidates when relevant.
- Use predefined inference rule short names only.

Response Format:

- `STUDENT_ERRORS`: Brief explanation or `Correct`
- `NEXT_STEP_CORRECTNESS`: `Correct` / `Suboptimal` / `Incorrect`
- `TEACHER_FEEDBACK`: Scaffolded guidance without answer revelation

Figure 16: Base System prompt for Teacher

Judge (Verifier) Prompt

Role: You are an expert pedagogical AI Judge for propositional logic proof problems, with access to the complete solution (`KNOWLEDGE_BASE_STEPS`). You evaluate both the student's proposed next step and the Teacher's feedback.

Task:

1. Compare the student's response against the knowledge-base solution.
2. Identify errors in the student's reasoning, if any.
3. Classify the student's next step as *Correct*, *Valid Alternative*, or *Incorrect*.
4. Evaluate whether the Teacher's feedback correctly guides the student.
5. Either enhance the Teacher's feedback or override it with corrected guidance.

Constraints:

- Do not reveal the exact next step, rule, or parent statements from the solution.
- Acknowledge correct aspects of the student's attempt before addressing errors.
- Use Socratic questions to guide reasoning; scaffold rather than instruct.
- Override Teacher feedback if it is incorrect, misleading, or reveals the solution.
- Keep final feedback concise (2–3 sentences) and encouraging.
- Use predefined inference rule short names only.

Response Format:

- `STUDENT_ERRORS`: Brief explanation or `Correct`
- `NEXT_STEP_CORRECTNESS`: `Correct` / `Suboptimal` / `Incorrect`
- `TEACHER_FEEDBACK_CORRECTNESS`: Assessment of Teacher feedback
- `JUDGE_ACTION`: `Enhanced` or `Overridden`
- `FINAL_FEEDBACK`: Judge-approved scaffolded guidance

Figure 17: Base System prompt for Judge

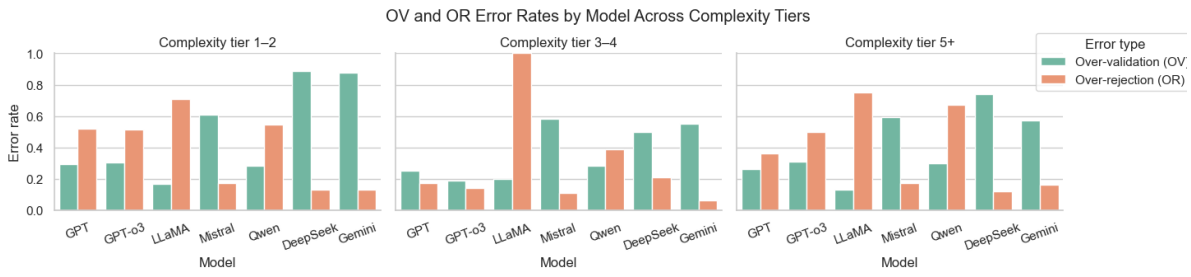


Figure 18: Over-validation (OV) and over-rejection (OR) rates by model across complexity tiers.

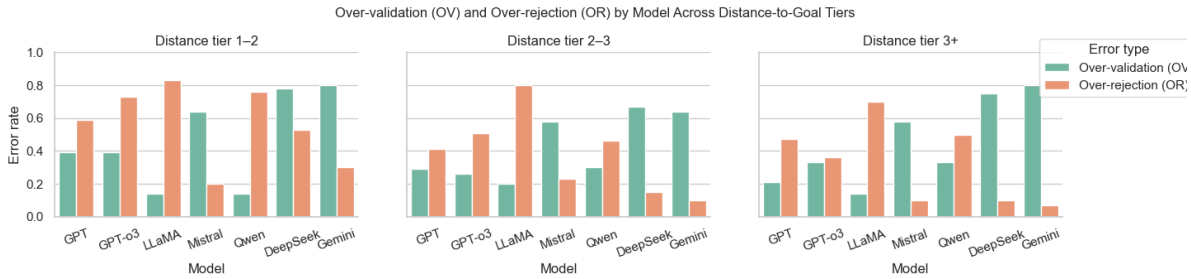


Figure 19: Over-validation (OV) and over-rejection (OR) rates by model across distance-to-goal tiers.

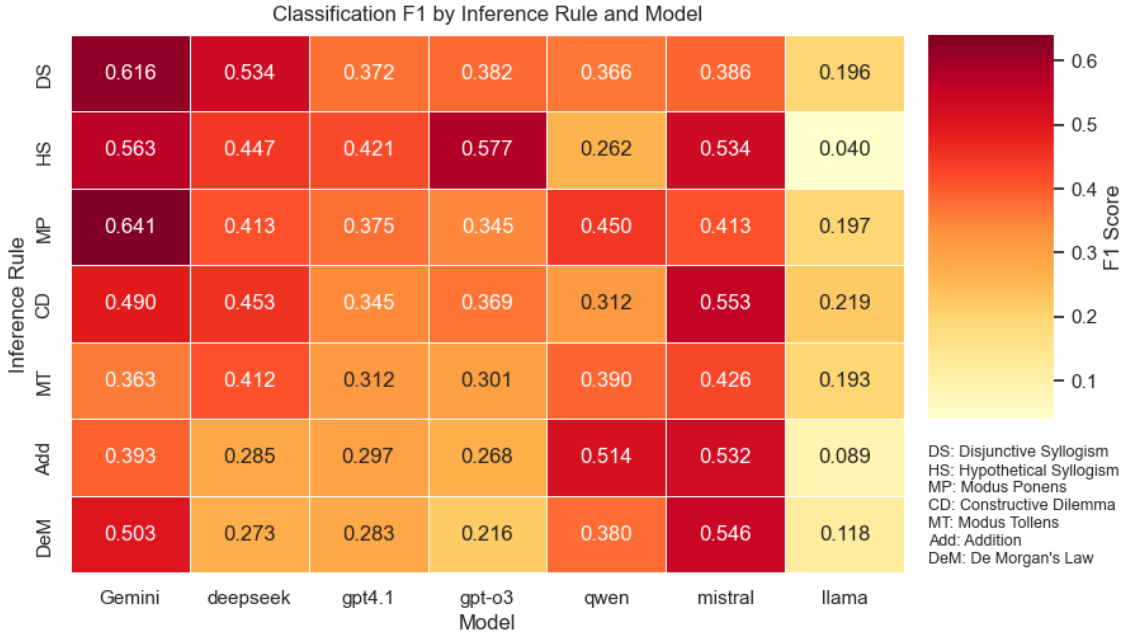


Figure 20: Classification F1 by inference rule and model. DS: Disjunctive Syllogism, HS: Hypothetical Syllogism, MP: Modus Ponens, CD: Constructive Dilemma, MT: Modus Tollens, Add: Addition, De Morgan's Law.