# Do Newer Models Make Better Tutors?: Characterizing Real-World Failures of AI Tutors in Higher Education Across Two Years, Two Prompts, and Five Models

Michael Tompkins
Arizona State University
Tempe, AZ, USA
mctompk1@asu.edu

Nihaarika Agarwal
Arizona State University
Tempe, AZ, USA
nagarw48@asu.edu

Robert Wasinger
Arizona State University
Tempe, AZ, USA
rwasinge@asu.edu

Connor Nelson
Arizona State University
Tempe, AZ, USA
connor.d.nelson@asu.edu

Daniel Votipka
Tufts University
Medford, MA, USA
dvotipka@cs.tufts.edu

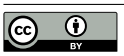
Yan Shoshitaishvili
Arizona State University
Tempe, AZ, USA
yans@asu.edu

Jaron Mink
Arizona State University
Tempe, AZ, USA
jaron.mink@asu.edu

## Abstract

While AI tutors are increasingly deployed in higher education alongside ever-evolving frontier model releases, they can also fail, providing students with incorrect or nonsensical answers that can drive disengagement or seed misconceptions. While prior work has begun to evaluate pedagogical risks via benchmarks, no work has characterized what failures tutors exhibit during real student interactions in upper-division higher-education coursework, or whether failures change across model updates.

To answer this, we study the failures of an AI tutor deployed for two years in a hands-on upper-division cybersecurity course on a freely accessible online platform. During this time, the AI tutor system was successively updated (1) from a non-reasoning to a reasoning model, (2) from a non-gated system prompt to an effort-gated system prompt, and (3) across three consecutive version releases of the same reasoning model. By coding AI tutor failures across 974 conversations involving 722 unique students, we investigate how AI tutors fail in practice within upper-division computing coursework. *First*, we show that failures are common, appearing in 34% of coded conversations, and fall into five failure modes: revealing solutions, withholding help, confidently providing wrong guidance, repeating the same advice, and misinterpreting student questions. *Second*, we find that while the introduction of a reasoning model reduced overall failures, all other prompt-based and model-version updates did not significantly reduce failure rates but instead shifted the distribution of failure modes. We end by discussing the implications of these results for future tutor designs, researchers, and course deployments.



This work is licensed under a Creative Commons Attribution 4.0 International License.
*SIGCSE TS 2027, Sacramento, CA, USA*
© 2027 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-3075-7/2027/02
https://doi.org/10.1145/3857222.3860125

## CCS Concepts

• **Social and professional topics → Computing education**.

## Keywords

AI tutors; large language models; Socratic tutoring; failure modes; computing education; qualitative coding

**ACM Reference Format:**
Michael Tompkins, Nihaarika Agarwal, Robert Wasinger, Connor Nelson, Daniel Votipka, Yan Shoshitaishvili, and Jaron Mink. 2027. Do Newer Models Make Better Tutors?: Characterizing Real-World Failures of AI Tutors in Higher Education Across Two Years, Two Prompts, and Five Models. In *Proceedings of the 58th ACM Technical Symposium on Computer Science Education V.1 (SIGCSE TS 2027), February 17–20, 2027, Sacramento, CA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3857222.3860125

## 1 Introduction

The introduction of Large Language Models (LLMs) makes it possible to offer a scalable, computer-enabled form of one-on-one tutoring, one of the most effective forms of instruction [6]. As such, computing courses are increasingly adopting and experimenting with AI tutors [21, 26, 27, 45].

Because unrestricted AI help can harm learning [5], AI tutors are often designed to be Socratic, guiding students toward answers without revealing them; however, LLM-based systems can still fail. AI tutors can give confidently wrong answers [19, 24, 25], generic replies that ignore what the student said [16, 28], or the same suggestion again and again [1, 28]. Such failures can disengage students [13] and leave them with misconceptions [25]. Existing work evaluates tutors on static benchmarks for general subjects [18, 28, 29, 31], but, to our knowledge, no work has examined how tutors fail with real students in upper-division cybersecurity courses.

Additionally, it is unclear whether or how these failures change alongside technological advances. As AI companies continuously release 'improved' models that promise better reasoning, accuracy, and responses, AI tutors can be updated to each new release, and

tutor developers can also revise a tutor's system prompt based on their in-course experience. However, newer models may not improve every behavior uniformly [7, 33] and can be more direct and less adaptive [29, 31], so updates may introduce new harms within tutoring.

Thus, to better understand real-world AI tutoring failures and how they vary across model updates, we ask:

**RQ1** How often do AI tutors fail within upper-division, higher-education coursework, and what failures do they exhibit?

**RQ2** How do changes in ❶ model type, ❷ prompting, and ❸ model version affect the frequency and distribution of these failures?

To answer these questions, we study an AI tutor deployed for two years in a hands-on, upper-division cybersecurity course on a free online platform. Over that time, ❶ its model switched from non-reasoning to reasoning (working through a problem step by step before replying), ❷ its prompt gained an effort gate (a rule to withhold help until a student shows their work), and ❸ its model went through three successive versions. By qualitatively coding tutor failures across 974 conversations from 722 students, we discover how AI tutors fail in practice. *First*, we find that failures are common (34% of coded conversations) and fall into five failure modes that mirror known failures of human tutors: revealing solutions too early, withholding help behind generic replies, giving confidently wrong guidance, repeating the same advice, and misreading student questions. *Second*, we find that model updates did not consistently reduce failures. Switching to a reasoning model (❶) cut the failure rate from 54.5% to 34.3% but shifted the most common failure from repeated advice to revealing solutions too early. Adding the effort gate (❷) did not significantly change the failure rate, and the three later versions (❸) brought no lasting improvement; generic replies rose significantly with both. Thus, while we did not see updates remove failures, they changed which failures occurred. We end by discussing implications for tutor evaluation and deployment.

| Model | Deployed | *n* | Fail. | % | 95% CI |
|---|---|---|---|---|---|
| gpt-4o | May '24 | 200 | 109 | **54.5** | [47.6, 61.3] |
| → *reasoning added* | | | | | |
| gpt-5 | Sep '25 | 204 | 70 | **34.3** | [28.1, 41.1] |
| → *effort gate added* | | | | | |
| gpt-5.1 | Nov '25 | 192 | 59 | **30.7** | [24.6, 37.6] |
| gpt-5.2 | Jan '26 | 184 | 36 | **19.6** | [14.5, 25.9] |
| gpt-5.4 | Mar '26 | 194 | 57 | **29.4** | [23.4, 36.1] |

**Table 1:** The five backing models in deployment order. Each served until the next was deployed. *n* counts coded conversations after exclusions; intervals are Wilson 95% CIs.

## 2 Related Work & Background

Deployed computing-education tutors often use system prompts as Socratic guardrails, telling the tutor to guide students rather than reveal solutions [21, 26, 27, 45]. Yet LLM programming help often falls short: LLM fixes for error messages were correct in only a third of cases [24], solutions leaked despite prompts against it [19], and students often missed a chatbot's errors, negatively impacting their learning [25]. Whether a tutor tells or guides also matters. In a randomized trial, students mastered more topics when their human tutors had an AI assistant that helped them ask guiding questions rather than give away answers [41]. System prompts are also an imperfect lever: models can over-apply rules, refusing even harmless requests [37], and in a CS1 study, adding pedagogical scaffolds to a tutor's system prompt brought no detectable benefit [4]. Model updates add risk too: a newer model can fail on cases an older one handled [14], and instruction following can drift between releases [7].

Human tutoring research explains what makes tutoring work, and so what failure looks like. One-on-one tutoring helps not because it delivers information [8] but because its guidance is *contingent* on what the student just did [40, 44]. Socratic questioning is defensible for the same reason [36], but only when its questions stay contingent on the student's state. Requiring effort before help also has roots in intelligent tutoring systems, where students often abuse on-demand hints [2, 3], yet withholding help by rule can backfire, whether set by prompt [38] or by training [12].

This tension is known as the assistance dilemma [23]: at every turn a tutor chooses between helping and withholding, and both carry risk. Too much help cuts short the productive struggle where learning happens [20], while too little, or help not tuned to the student's state, often fails and can even hurt performance [17, 22]. Good tutoring also has requirements outside this dilemma: tutor and student must keep common ground on what is being attempted [9], the tutor must adapt to the student's responses [8, 44], and its explanations must be correct.

Existing evaluations of AI tutors mostly score abilities or check for risks chosen in advance rather than found in the data, using tutoring corpora turned into benchmarks [29, 31], real conversations scored on fixed dimensions [28], or generated dialogues built to provoke known harms [18]. Others study a single failure, such as answer leakage [46]. Closest to our setting, CodeAid deployed a guarded tutor in a large course and reported overall response quality before and after a model upgrade [21]. Thus, while prior work has begun to audit AI tutors against predefined risks and to report overall quality from live deployments, to our knowledge, no work has examined which failures emerge from a deployed tutor's real conversations and how they shift as its model changes.

## 3 Methods

To study how AI tutors fail in practice, we analyze two years (May 2024–May 2026) of real-world student–AI tutor interactions within an open-access cybersecurity curriculum and across five successive models.

**Course Deployment.** We analyze conversations from pwn.college, a free educational platform where students can watch lectures and complete hands-on cybersecurity exercises in a virtualized Linux environment in their browser. This curriculum also serves as an upper-division cybersecurity course at a state university and is intended for third-year Computer Science students. The curriculum spans ten cybersecurity modules (e.g., Web Security, Cryptography, Reverse Engineering), each consisting of hands-on challenges in which students apply the concepts taught to retrieve and submit

a secret flag. For example, in the web security module, students perform an SQL injection to extract the flag from a database. At any point in a challenge, students can receive help from SENSAI [34], an integrated AI tutor, in a separate chat-based interface.

**Tutor Design and Updates.** When a student submits a query to the AI tutor, the tutor bundles it into a templated request combining (1) a system prompt instructing the model to act as a Socratic mentor that guides students toward solutions without revealing them, (2) the conversation history, (3) the student's new message, and (4) the student's current working context (their latest saved file and recent terminal history, i.e., the terminal screen plus scrollback). The bundled request is sent to the backing model, and its response is returned to the chat. The tutor is not given the solution to the challenge. A conversation lasts from when the student opens the tutor until they reload the page or restart the challenge. Throughout the deployment, the backing model was routinely updated to the latest release while the system prompt was updated once; this gives five distinct measurement periods, one per model (Table 1). We ran gpt-4o at temperature 0.4 and the gpt-5-series models at medium reasoning effort. Under our original prompt, which was deployed for gpt-4o and gpt-5, the tutor received short Socratic instructions to ask probing questions and avoid giving away answers. Then, we iterated on the prompt to introduce a new *effort gate* along with explicit limits on what the tutor may do, deploying the revised prompt from gpt-5.1 onward.[1] We added the gate because gpt-5 often gave solutions outright when students showed minimal progress, and we wanted to make help more contingent on demonstrated work [43]. Under the gate, the tutor first judges whether a student's question is "thoughtful," which the prompt defines as stating the goal, what the student tried, what happened, and why they think it failed. If it is not, the tutor is instructed to reply with three questions: "What did you try? What happened? Why do you think it didn't work?"

**Recruitment and Collected Data.** Students opted into the AI tutor by giving informed consent to research logging and use of their platform interactions. For each conversation, we collected the student's messages, tutor's replies, and the context attached to each query, along with metadata such as the backing model version and module number. We collected data continuously throughout the tutor's deployment, accumulating over 200,000 logged conversations. From this full dataset, we drew a random sample stratified by backing model and course module, targeting twenty conversations per model × module pair across five models and ten modules, resulting in 1,015 conversations.[2] We removed 40 non-English conversations and one cut off mid-conversation by a system update, leaving 974 conversations from 722 students. We did not draw replacements, leaving 184 to 204 conversations per model; because we compare per-model rates, this slightly reduces precision but does not bias the comparisons.

**Qualitative Analysis.** We built the failure codebook through iterative open coding; two of the three coders had been teaching assistants for the course. Coders read each conversation in full, since failures can span several turns. We define a *failure* broadly to include not only factually incorrect information but also any tutor reply that works against the student's learning in context, including replies that are accurate or reasonable in isolation, such as a generic Socratic question, when they ignore what the student has already shown. We assigned one code per conversation, so each failed conversation counts toward exactly one failure mode, as the distribution comparisons in Section 5 require. When multiple failures appeared, coders applied a precedence order that emerged during open coding, ranking the modes (Section 4) by likely harm to the student (*incorrect guidance* > *stuck loop* > *misread student turn* > *premature solution* > *deflect to generic*), since wrong guidance can go uncaught [25], stuck loops leave students at an impasse [13], misread turns target the wrong problem [9], and premature solutions and deflections give too much or too little help [23].

To build the initial codebook, two authors (A and B) inductively coded stratified random samples of 50 conversations per round; after three rounds (150 conversations), no new codes emerged. To establish reliability, two coders then independently coded new 50-conversation samples per round, computing Cohen's $\kappa$ [32] and resolving disagreements after each round until $\kappa \geq 0.8$. Coders A and B reached $\kappa = 0.804$ after four rounds, and a third coder (C) later reached $\kappa = 0.830$ after two. Coders A, B, and C then single-coded all 974 conversations.

**Quantitative Analysis.** We run three comparisons across the five models: ❶ the move to a reasoning model (gpt-4o vs. gpt-5); ❷ the effort-gated prompt (gpt-5 vs. gpt-5.1, which also changed the model); and ❸ successive model versions (gpt-5.1, gpt-5.2, and gpt-5.4). We detail our analysis in Section 5.

**Ethical Considerations.** Our institution's IRB approved this study. Participation was voluntary and limited to adults; students could opt out at any time, and their choice did not affect their grades. The tutor was optional, and course staff remained available. The course and challenge materials given to the backing models are our own, and the IRB approved this use. Student messages reached the model provider only through the tutor's normal operation, with model training disabled. Coders saw no student identifiers, and we report aggregate results and quote only short, non-identifying excerpts.

**Limitations.** *First,* we measure instructional risk through coded failure modes rather than learning gains, so we limit our claims to tutor behavior. Coders also saw only the logs, not students' off-platform work, earlier attempts, or thinking time. *Second,* the study is observational with no random assignment or control. The prompt change came with a model update, so that comparison (❷) cannot separate the two and we treat it as descriptive. We rely more on the other two comparisons, where the prompt stayed the same. Even those span months in which the student population may have changed, especially gpt-4o's period of over a year. Non-English exclusions, for example, rose from none under gpt-4o to 15 under gpt-5.2. Long conversation history may also contribute to failures, though 34 of the 64 deflections where coders noted the failing turn came in the tutor's first reply. *Third,* we studied one course, English-language conversations only, and self-selected students who may be more comfortable with AI than most. Our two prompts were written by course staff rather than systematically optimized, so the failures we report may partly reflect those particular prompts. We also used only OpenAI's GPT models, so our claims primarily concern this

[1] Both prompts are available at https://github.com/frqmod/SIGCSE_SENSAI_prompts.

[2] A module-labeling error added 15 conversations to three cells (gpt-5 modules 8 and 9, gpt-5.4 module 9). Module-adjusted odds ratios from a Cochran–Mantel–Haenszel test [10, 30] and a logistic regression with module as a covariate closely match the unadjusted ones in every comparison (e.g., 0.45 and 0.43 vs. 0.44 for the reasoning model), so the imbalance does not bias our estimates.

model family rather than LLMs in general. Still, these failure modes are domain-neutral behaviors: similar ones appear in other domains and tools [18, 28, 29], and GPT models back nearly every deployed tutor cited in Section 2. *Fourth,* some students contributed several conversations, and because a student's habits can influence whether a failure occurs, these repeat conversations may make our results look slightly more certain than they are. Coding one failure per conversation also undercounts co-occurring failures, though we retained the one we ranked most harmful.

## 4 How Do AI Tutors Fail? (RQ1)

Of the 974 sampled conversations, 331 (34.0%, Wilson 95% CI [31.1%, 37.0%]) contained at least one tutor failure. Failures occurred in every module and under every model.

Failures fall into five modes (Table 2), described below. The two most common, *premature solution* and *deflect to generic*, account for 62% of failures and sit on opposite sides of the assistance dilemma, either giving too much help or too little. The other 38% of failures (*incorrect guidance*, *stuck loop*, and *misread student turn*) are breakdowns of the interaction itself. These failures parallel documented issues in human tutoring (Section 6).

**Premature Solution.** In 133 conversations (13.7% of conversations, 40.2% of failures), the tutor revealed a solution or key step before the student had worked through it, short-circuiting the productive struggle its Socratic system prompt aims to preserve. A reply is considered a premature solution if the student could take the next step without any reasoning of their own, as when the tutor provides a fully working command or a specific value such as a password; a template with placeholders, a conceptual explanation, or a syntax example is not. In one case, a challenge asked students to write a script that fills up the disk. A student's script printed a few lines and then stopped with an error, and the student reported this to the tutor. The tutor explained exactly why the script stopped and handed over a corrected script that solved the challenge, instead of helping the student work out what went wrong.

**Deflect to Generic.** In 71 conversations (7.3% of conversations, 21.5% of failures), the tutor withheld contingent help, giving a generic reply that ignored what the student said. Generic questions are not failures in themselves; to count, the reply must ignore a specific, answerable question or the evidence the student shared. Several turns of generic check-in questions with no concrete pointer also count, as do replies that only direct the student to alternative help channels (e.g., a human TA or the course Discord). In one case, a student attacking an encryption scheme had shared their script, its output, and the error it raised, and had received detailed help in return. When they then asked a focused question about one step of the attack, the tutor ignored everything they had shared and replied only with a generic "What did you try? What happened? Why do you think it didn't work?"

**Incorrect Guidance.** In 65 conversations (6.7% of conversations, 19.6% of failures), the tutor gave a confidently wrong hint, explanation, or diagnosis. Only confident claims that would lead the student to a wrong result count, not hedged guesses or valid alternative approaches. For example, a challenge required entering a password in a specific text encoding (UTF-16), and a student's attempts kept being rejected. The tutor confidently supplied a shell command to write the password's exact bytes (`printf $'\xFF\xFEz\x00...' > [file]`), but the shell cuts the string off at the first null byte, so the command wrote only three of the eighteen bytes needed and could not work. Novices tend to over-trust AI assistants [25] and may not catch such errors.

**Stuck Loop.** In 54 conversations (5.5% of conversations, 16.3% of failures), the tutor repeated unadapted guidance that kept the interaction from moving forward even as the student tried to make progress. Cases such as repeating a reply nearly word for word, cycling among a few suggestions, and restarting from the basics after progress all count, but only when the student has added new information that the tutor ignores. In one case, a student in a web challenge was trying to work out which address a page should redirect to. After they reported that both addresses they had tried were wrong, the tutor repeated its previous reply almost word for word: a checklist for fixing a missing-page (404) error that did not address their question.

**Misread Student Turn.** The rarest failure mode was *misread student turn* (8 conversations: 0.8% of conversations, 2.4% of failures), in which the AI tutor completely misinterpreted what the student said or did. Unlike with *incorrect guidance*, the tutor's reply could have been right elsewhere; it addressed a situation the student was not in, as when the tutor missed a reported success, claimed one that did not happen, or invented an error. In one case, a student's script for recovering the flag printed a visibly garbled one ("pwn.csllege" instead of the expected "pwn.college"), yet the tutor told them the script had worked and recovered the whole flag.

## 5 Do Model Updates Affect Failures? (RQ2)

We now examine how failures changed with ❶ the move to a reasoning model, ❷ the effort-gated prompt, and ❸ successive model versions.

*Overall Failure Rate:* We compare adjacent models' failure and non-failure counts with Fisher's exact test [15], which we use for all pairwise contrasts because several per-mode tables have near-empty cells where the $\chi^2$ approximation breaks down. We report each model's failure rate with a Wilson 95% confidence interval [42] and each difference as an odds ratio (OR) with its 95% confidence interval (Table 3). For the three-model comparison, we first run an omnibus $\chi^2$ test, then Bonferroni-correct its three pairwise contrasts. We also compare the newest model, gpt-5.4, with gpt-4o and gpt-5 post hoc, using the same tests.

*Failure Distribution:* To test whether the mix of failures changed, we take only the failed conversations and run an omnibus $\chi^2$ test on the table of failure modes by model.[3] We report Cramér's $V$ as the effect size [11] (0 means identical mixes; 1 means no overlap). Because some modes are rare, we take each $p$-value from an exact permutation test. When the omnibus test is significant, we test each mode against all other failures (Fisher's exact for two models, permutation-$\chi^2$ for three), applying a Bonferroni correction across the five modes. Because a mode's share can rise just because other modes shrink, we also consider each mode's per-conversation rate.

**❶ Reasoning reduced failures, non-uniformly.** Comparing gpt-4o with gpt-5 isolates the model change, since the system prompt stayed the same. The failure rate dropped significantly, from 54.5%

[3] We use $\chi^2$ because Fisher's exact test is typically limited to 2×2 tables.

| Failure mode | Description | *n* | % conv. | % fail. |
|---|---|---|---|---|
| **Assistance Dilemmas** | | | | |
| Premature solution | reveals a solution or key step before the student has worked it out | 133 | 13.7 | 40.2 |
| Deflect to generic | withholds help; generic replies that ignore the student's context | 71 | 7.3 | 21.5 |
| **Interactional Failures** | | | | |
| Incorrect guidance | gives a confidently wrong hint, explanation, or diagnosis | 65 | 6.7 | 19.6 |
| Stuck loop | repeats unadapted guidance without moving the interaction forward | 54 | 5.5 | 16.3 |
| Misread student turn | completely misinterprets what the student said or did | 8 | 0.8 | 2.4 |

**Table 2:** The five failure modes (RQ1), with each mode's share of all coded conversations and of failures.

| Comparison | OR [CI] | *p* |
|---|---|---|
| **❶ Reasoning Model (4o→5)** | **0.44 [0.29, 0.65]** | **<0.001** |
| ❷ Prompt Change (5→5.1) | 0.85 [0.56, 1.29] | 0.455 |
| **❸ Model Versions (5.1/5.2/5.4)** | **omnibus $\chi^2(2) = 7.10$** | **0.029** |
| gpt-5.1→gpt-5.2 | 0.55 [0.34, 0.89] | $0.052^{\dagger}$ |
| gpt-5.1→gpt-5.4 | 0.94 [0.61, 1.45] | $>0.99^{\dagger}$ |
| gpt-5.2→gpt-5.4 | 1.70 [1.06, 2.74] | $0.094^{\dagger}$ |

**Table 3: Comparison of Failures Across Models.** Pairwise rows report Fisher's exact tests and odds ratios. The model-versions row reports the omnibus $\chi^2$ across its three versions. OR < 1 means fewer failures for the later model. Bold designates significance at $\alpha = 0.05$. $^{\dagger}$Bonferroni-corrected ($m = 3$); CIs are unadjusted.

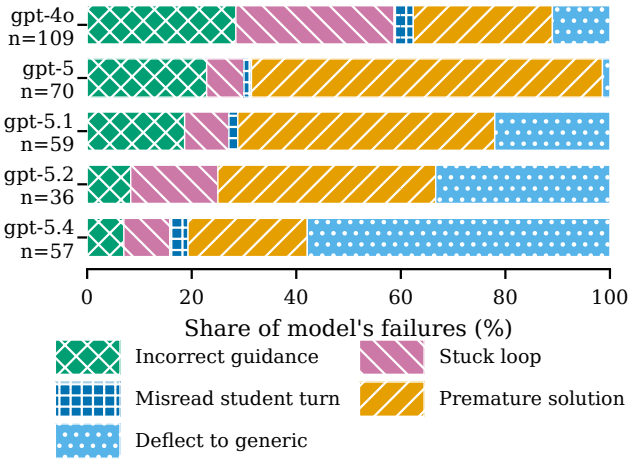



**Figure 1:** The mix of failure modes per backing model: each bar splits that model's failures across the five modes (axis labels give each model's failure count; overall rates appear in Table 1). Modes are ordered left to right by coding precedence (Section 3).

to 34.3% ($p < 0.001$), and the mix of failures also changed (Figure 1; Cramér's $V = 0.44$, $p < 0.001$). *Stuck loop* fell from 30% to 7% of failures while *premature solution* rose from 27% to 67% (both $p < 0.001$). The reasoning model cut repetitive help, but revealing solutions too early became more common.

The transcripts show this trade-off. Under gpt-4o, students reported that a suggestion failed, only to get the same suggestion again, such as in one decoding challenge where a student replied "I implemented it and was hit with a non-hexadecimal number found in fromhex() error," and the tutor restated the same fix that resulted in the error. Under gpt-5, the tutor was more capable overall but also gave away more: when a student in a web security challenge asked "What should the payload look like?", the tutor described the exact payload and gave an example instead of explaining the input conditions or how to find them.

**❷ With the effort gate, failures did not fall but their mix shifted.** We next compare gpt-5 with gpt-5.1, whose revised prompt added the effort gate (Section 3) in response to these premature solutions. Because the prompt change came with a model update, the comparison cannot separate the two, so we read it descriptively.

The failure rate did not change significantly (34.3% to 30.7%, $p = 0.455$), but the mix of failures did (Cramér's $V = 0.34$, $p = 0.003$). Only *deflect to generic* changed significantly, rising from 1% to 22% of failures ($p = 0.001$); *premature solution* fell from 67% to 49%, but not significantly ($p = 0.24$). This is the assistance dilemma in action: the same over- versus under-helping trade-off appears when withholding is imposed through a tutor's training objective [12], and withholding imposed by prompt has frustrated students elsewhere [38].

The gpt-5.1 transcripts show both sides of the rule. It often worked as intended: when students asked for the solution with no sign of effort (e.g., "i dont know what im supposed to do tell me how to solve it please"), the tutor replied with the three questions. However, it also withheld help from students who had already shown their work: one student deep in a man-in-the-middle challenge pasted their script and asked for help with a minor adjustment, and the tutor returned the same three questions. The sticking point was often the third question: even students who came with a goal and an attempt got no help until they also explained what they thought went wrong.

**❸ Newer versions did not reliably improve the tutor.** Finally, we compare the three versions that ran under the revised prompt (gpt-5.1, -5.2, and -5.4). The failure rate moved without a clear trend (30.7%, 19.6%, 29.4%; omnibus $p = 0.029$), and no single version change stayed significant after Bonferroni correction (gpt-5.1 to -5.2: $p = 0.017$, corrected $p = 0.052$). The mix of failures, however, shifted significantly (Cramér's $V = 0.27$, $p = 0.003$), driven largely by *deflect to generic*, which rose from 22% to 58% of failures ($p = 0.001$). This fits evidence that added prompt constraints can degrade even advanced models on the underlying task [35], that instruction

following can change between releases months apart [7], and that larger models do not improve uniformly and can do worse when an easier task can be confused with the harder real one [33].

Across all three versions, the tutor still returned the three questions to students who asked for help without showing any work ("not sure how to proceed," "how do i get the flag?"). What grew was the tutor returning the same three questions to students who had already shown progress, over-applying the rule. *Deflect to generic* held near 7% of conversations under gpt-5.1 and gpt-5.2 (6.8% and 6.5%), then more than doubled to 17.0% under gpt-5.4. In one gpt-5.4 case, a student shared their script and its 404 error and asked what was wrong with the hostname; the tutor replied that it could "see the script and the 404" but still asked only the three questions. Because the prompt did not change across these versions, the increase most likely comes from the model, though we cannot rule out drift in the student population. Across the whole deployment, the newest model, gpt-5.4, failed less often than gpt-4o (29.4% vs. 54.5%; OR = 0.35, $p < 0.001$) but not significantly less often than the first reasoning model, gpt-5 (34.3%; $p = 0.33$). Its failures also differed in kind: mostly generic deflections, where gpt-5's were mostly premature solutions (Cramér's $V = 0.67$, $p < 0.001$).

## 6 Discussion and Future Work

Overall, model updates changed which failures the tutor made (Section 5). Only the switch to a reasoning model significantly lowered the failure rate (❶; Table 3), but all three comparisons showed a significant change in the mix of failures (Figure 1). Even the newest model did not fail significantly less often than the first reasoning model, only differently. The reasoning model's gains came with more premature solutions: a model better at solving problems may also be quicker to solve them for the student. One explanation for these shifts is that more capable models tend to be more direct and less adaptive [29, 31]: later models may have applied the same instructions more aggressively, giving less contingent replies over time. The rise in deflection also resembles over-refusal, where models over-apply safety rules and refuse even harmless requests [37]. Prompt rules can also simply have no effect: a CS1 study found no detectable benefit from adding pedagogical scaffolds to a tutor's prompt [4]. Either way, a prompt rule is something the model interprets, not a guarantee.

The failures we found (Section 4) also mirror known failure modes in human tutoring. The two most common, *premature solution* and *deflect to generic*, correspond to the two sides of the assistance dilemma human tutors face in balancing over- and under-helping [23]. The other three mirror breakdowns known from tutoring and feedback research: wrong explanations that learners often miss, to the detriment of their learning [25] (*incorrect guidance*); guidance that does not adapt to the student, which makes tutoring less effective [44] (*stuck loop*); and misreading what the student is attempting, so guidance ends up targeting a problem the student does not have [9, 39] (*misread student turn*). Thus, these failures are not unique to AI. Without a human baseline, we cannot say whether AI tutors fail more often than humans, but rates as high as 55% (gpt-4o; Table 1) expose many students to these harms.

**Advice for Tutor Design.** Our guardrails traded one failure for another rather than removing failures. The effort gate seemed to work at first, with somewhat fewer premature solutions, but *deflect to generic* rose and then more than doubled under gpt-5.4. Help works best when it builds on what the student has already shown [17], yet the model often applied the rule too literally, deflecting students who had already shared their progress. Designers should therefore check whether a guardrail creates new failures, not only whether it stops the one it targets. Instead of an all-or-nothing rule, they might use hints that start general and get more specific as the student keeps working. Model updates also change a tutor's failures without any prompt change: *deflect to generic* jumped from 6.5% to 17.0% of conversations between gpt-5.2 and gpt-5.4. Prompts and their failures should therefore be re-audited whenever the backing model changes, for example by replaying a sample of past conversations on the new model and comparing failure rates and mixes before students use it.

**Advice for Educators.** Educators adopting tutors should watch for quiet failures in addition to obvious ones. Our tutor's design, which aimed to avoid revealing solutions too early, came with more *deflect to generic* failures, systematically denying students help. Some of this harm may also be reduced without technical changes, for example by telling students how to avoid specific failures (e.g., explaining their reasoning so replies are not generic).

**Advice for Researchers.** Our results caution against one-time evaluations: each model update produced different failure rates and mixes, so researchers should not assume failures are static or that newer models will reduce them. From gpt-5.1 to gpt-5.4, for example, the failure rate ended up similar, but the mix of failures changed significantly. Evaluations should measure both how often a tutor fails and how, and our taxonomy can support this. Future work should also study how students respond to and recover from these failures, and how the failures affect learning outcomes.

## 7 Conclusion

We studied an AI tutor's failures over two years of deployment in upper-division cybersecurity coursework, coding 974 conversations with 722 students. First, 34% of coded conversations contained at least one of five types of tutor failure. Second, model updates did not consistently reduce or remove these failures: switching to a reasoning model reduced failures, but the other prompt and model updates did not significantly reduce failure rates and instead changed which failures occurred. We encourage designers and researchers to build and evaluate tutors with model updates in mind, and educators to consider how these failures affect their teaching and which non-technical interventions may reduce them.

## Acknowledgments

This material is based upon work supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. HR001124C0362. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Defense Advanced Research Projects Agency (DARPA).

## References


[1] Erfan Al-Hossami, Razvan Bunescu, Justin Smith, and Ryan Teehan. 2024. Can Language Models Employ the Socratic Method? Experiments with Code Debugging. In *Proceedings of SIGCSE TS 2024*. ACM, 53–59. doi:10.1145/3626252.3630799

[2] Vincent Aleven, Elmar Stahl, Silke Schworm, Frank Fischer, and Raven Wallace. 2003. Help Seeking and Help Design in Interactive Learning Environments. *Review of Educational Research* 73, 3 (2003), 277–320. doi:10.3102/00346543073003277

[3] Ryan Shaun Baker, Albert T. Corbett, Kenneth R. Koedinger, and Angela Z. Wagner. 2004. Off-Task Behavior in the Cognitive Tutor Classroom: When Students "Game the System". In *Proceedings of CHI 2004*. ACM, 383–390. doi:10.1145/985692.985741

[4] Maximilian Georg Barth, Sverrir Thorgeirsson, Khashayar Etemadi, Juho Leinonen, Carlos Cotrini, and Zhendong Su. 2026. Steering AI Tutors Through System Prompts: A Crossover Study on Self-Regulated Learning and Cognitive Engagement Scaffolds in CS1. In *Proceedings of ICER 2026*. ACM, 104–118. doi:10.1145/3765964.3811656

[5] Hamsa Bastani, Osbert Bastani, Alp Sungu, Haosen Ge, Ozge Kabakci, and Rei Mariman. 2025. Generative AI without guardrails can harm learning: Evidence from high school mathematics. *Proceedings of the National Academy of Sciences* 122, 26 (2025). doi:10.1073/pnas.2422633122

[6] Benjamin S. Bloom. 1984. The 2 Sigma Problem: The Search for Methods of Group Instruction as Effective as One-to-One Tutoring. *Educational Researcher* 13, 6 (1984), 4–16. doi:10.3102/0013189X013006004

[7] Lingjiao Chen, Matei Zaharia, and James Zou. 2024. How Is ChatGPT's Behavior Changing over Time? *Harvard Data Science Review* 6, 2 (2024). doi:10.1162/99608f92.5317da47

[8] Michelene T. H. Chi, Stephanie A. Siler, Heisawn Jeong, Takashi Yamauchi, and Robert G. Hausmann. 2001. Learning from Human Tutoring. *Cognitive Science* 25, 4 (2001), 471–533. doi:10.1207/s15516709cog2504_1

[9] Herbert H. Clark and Susan E. Brennan. 1991. Grounding in Communication. In *Perspectives on Socially Shared Cognition*. American Psychological Association, 127–149. doi:10.1037/10096-006

[10] William G. Cochran. 1954. Some Methods for Strengthening the Common $\chi^2$ Tests. *Biometrics* 10, 4 (1954), 417–451. doi:10.2307/3001616

[11] Harald Cramér. 1946. *Mathematical Methods of Statistics*. Princeton University Press, Princeton, NJ. doi:10.1515/9781400883868

[12] David Dinucu-Jianu, Jakub Macina, Nico Daheim, Ido Hakimi, Iryna Gurevych, and Mrinmaya Sachan. 2025. From Problem-Solving to Teaching Problem-Solving: Aligning LLMs with Pedagogy using Reinforcement Learning. In *Proceedings of EMNLP 2025*. 272–292. doi:10.18653/v1/2025.emnlp-main.15

[13] Sidney D'Mello and Art Graesser. 2012. Dynamics of Affective States During Complex Learning. *Learning and Instruction* 22, 2 (2012), 145–157. doi:10.1016/j.learninstruc.2011.10.001

[14] Jessica Echterhoff, Fartash Faghri, Raviteja Vemulapalli, Ting-Yao Hu, Chun-Liang Li, Oncel Tuzel, and Hadi Pouransari. 2024. MUSCLE: A Model Update Strategy for Compatible LLM Evolution. In *Findings of EMNLP 2024*. 7320–7332. doi:10.18653/v1/2024.findings-emnlp.430

[15] Ronald A. Fisher. 1922. On the Interpretation of $\chi^2$ from Contingency Tables, and the Calculation of P. *Journal of the Royal Statistical Society* 85, 1 (1922), 87–94. doi:10.2307/2340521

[16] Eduard Frankford, Clemens Sauerwein, Patrick Bassner, Stephan Krusche, and Ruth Breu. 2024. AI-Tutoring in Software Engineering Education. In *Proceedings of ICSE-SEET 2024*. ACM, 309–319. doi:10.1145/3639474.3640061

[17] John Hattie and Helen Timperley. 2007. The Power of Feedback. *Review of Educational Research* 77, 1 (2007), 81–112. doi:10.3102/003465430298487

[18] Rima Hazra, Bikram Ghuku, Ilona Marchenko, Yaroslava Tokarieva, Sayan Layek, Somnath Banerjee, Julia Stoyanovich, and Mykola Pechenizkiy. 2026. SafeTutors: Benchmarking Pedagogical Safety in AI Tutoring Systems. In *Findings of EMNLP 2026*. 40864–40896. doi:10.18653/2026.findings-emnlp.1917

[19] Arto Hellas, Juho Leinonen, Sami Sarsa, Charles Koutcheme, Lilja Kujanpää, and Juha Sorva. 2023. Exploring the Responses of Large Language Models to Beginner Programmers' Help Requests. In *Proceedings of ICER 2023*. ACM, 93–105. doi:10.1145/3568813.3600139

[20] Manu Kapur. 2008. Productive Failure. *Cognition and Instruction* 26, 3 (2008), 379–424. doi:10.1080/07370000802212669

[21] Majeed Kazemitabaar, Runlong Ye, Xiaoning Wang, Austin Z. Henley, Paul Denny, Michelle Craig, and Tovi Grossman. 2024. CodeAid: Evaluating a Classroom Deployment of an LLM-based Programming Assistant that Balances Student and Educator Needs. In *Proceedings of CHI 2024*. ACM, Article 650, 20 pages. doi:10.1145/3613904.3642773

[22] Avraham N. Kluger and Angelo DeNisi. 1996. The Effects of Feedback Interventions on Performance: A Historical Review, a Meta-analysis, and a Preliminary Feedback Intervention Theory. *Psychological Bulletin* 119, 2 (1996), 254–284. doi:10.1037/0033-2909.119.2.254

[23] Kenneth R. Koedinger and Vincent Aleven. 2007. Exploring the Assistance Dilemma in Experiments with Cognitive Tutors. *Educational Psychology Review* 19, 3 (2007), 239–264. doi:10.1007/s10648-007-9049-0

[24] Juho Leinonen, Arto Hellas, Sami Sarsa, Brent Reeves, Paul Denny, James Prather, and Brett A. Becker. 2023. Using Large Language Models to Enhance Programming Error Messages. In *Proceedings of SIGCSE TS 2023*. ACM, 563–569. doi:10.1145/3545945.3569770

[25] Tiffany Wenting Li, Yifan Song, Hari Sundaram, and Karrie Karahalios. 2025. Can Learners Navigate Imperfect Generative Pedagogical Chatbots? An Analysis of Chatbot Errors on Learning. In *Proceedings of L@S 2025*. ACM, 151–163. doi:10.1145/3698205.3729550

[26] Mark Liffiton, Brad E. Sheese, Jaromir Savelka, and Paul Denny. 2023. CodeHelp: Using Large Language Models with Guardrails for Scalable Support in Programming Classes. In *Proceedings of Koli Calling 2023*. ACM, Article 8, 11 pages. doi:10.1145/3631802.3631830

[27] Rongxin Liu, Carter Zenke, Charlie Liu, Andrew Holmes, Patrick Thornton, and David J. Malan. 2024. Teaching CS50 with AI: Leveraging Generative Artificial Intelligence in Computer Science Education. In *Proceedings of SIGCSE TS 2024*. ACM, 750–756. doi:10.1145/3626252.3630938

[28] Ying Liu, Can Li, Ting Zhang, Mei Wang, Qiannan Zhu, Jian Li, and Hua Huang. 2026. Discerning Minds or Generic Tutors? Evaluating Instructional Guidance Capabilities in Socratic LLMs. In *Findings of EMNLP 2026*. 33802–33834. doi:10.18653/2026.findings-emnlp.1588

[29] Jakub Macina, Nico Daheim, Ido Hakimi, Manu Kapur, Iryna Gurevych, and Mrinmaya Sachan. 2025. MathTutorBench: A Benchmark for Measuring Open-ended Pedagogical Capabilities of LLM Tutors. In *Proceedings of EMNLP 2025*. 204–221. doi:10.18653/v1/2025.emnlp-main.11

[30] Nathan Mantel and William Haenszel. 1959. Statistical Aspects of the Analysis of Data From Retrospective Studies of Disease. *Journal of the National Cancer Institute* 22, 4 (1959), 719–748. doi:10.1093/jnci/22.4.719

[31] 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 NAACL 2025*. 1234–1251. doi:10.18653/v1/2025.naacl-long.57

[32] Mary L. McHugh. 2012. Interrater reliability: the kappa statistic. *Biochemia Medica* 22, 3 (2012), 276–282. doi:10.11613/bm.2012.031

[33] Ian R. McKenzie et al. 2023. Inverse Scaling: When Bigger Isn't Better. *Transactions on Machine Learning Research* (2023). arXiv:2306.09479

[34] Connor Nelson, Adam Doupé, and Yan Shoshitaishvili. 2025. SENSAI: Large Language Models as Applied Cybersecurity Tutors. In *Proceedings of SIGCSE TS 2025*. ACM, 833–839. doi:10.1145/3641554.3701801

[35] Yunjia Qi, Hao Peng, Xintong Shi, Amy Xin, Xiaozhi Wang, Bin Xu, Lei Hou, and Juanzi Li. 2026. On the Paradoxical Interference between Instruction-Following and Task Solving. arXiv:2601.22047 [cs.CL]

[36] Carolyn Penstein Rosé, Johanna D. Moore, Kurt VanLehn, and David Allbritton. 2001. A Comparative Evaluation of Socratic versus Didactic Tutoring. In *Proceedings of CogSci 2001*. 869–874. https://escholarship.org/uc/item/98j4479r

[37] Paul Röttger, Hannah Rose Kirk, Bertie Vidgen, Giuseppe Attanasio, Federico Bianchi, and Dirk Hovy. 2024. XSTest: A Test Suite for Identifying Exaggerated Safety Behaviours in Large Language Models. In *Proceedings of NAACL 2024*. 5377–5400. doi:10.18653/v1/2024.naacl-long.301

[38] Sebastian Steindl, Fabian Brunner, Nada Sissouno, Dominik Schwagerl, Florian Schöler-Niewiera, and Ulrich Schäfer. 2025. On the Effectiveness of Prompt-Moderated LLMs for Math Tutoring at the Tertiary Level. In *Findings of EMNLP 2025*. 11310–11323. doi:10.18653/v1/2025.findings-emnlp.605

[39] Kurt VanLehn. 2006. The Behavior of Tutoring Systems. *International Journal of Artificial Intelligence in Education* 16, 3 (2006), 227–265. doi:10.3233/irg-2006-16(3)02

[40] Kurt VanLehn. 2011. The Relative Effectiveness of Human Tutoring, Intelligent Tutoring Systems, and Other Tutoring Systems. *Educational Psychologist* 46, 4 (2011), 197–221. doi:10.1080/00461520.2011.611369

[41] Rose E. Wang, Ana T. Ribeiro, Carly D. Robinson, Susanna Loeb, and Dora Demszky. 2024. Tutor CoPilot: A Human-AI Approach for Scaling Real-Time Expertise. arXiv:2410.03017 [cs.CY]

[42] Edwin B. Wilson. 1927. Probable Inference, the Law of Succession, and Statistical Inference. *J. Amer. Statist. Assoc.* 22, 158 (1927), 209–212. doi:10.1080/01621459.1927.10502953

[43] David Wood, Jerome S. Bruner, and Gail Ross. 1976. The Role of Tutoring in Problem Solving. *Journal of Child Psychology and Psychiatry* 17, 2 (1976), 89–100. doi:10.1111/j.1469-7610.1976.tb00381.x

[44] David Wood and David Middleton. 1975. A Study of Assisted Problem-Solving. *British Journal of Psychology* 66, 2 (1975), 181–191. doi:10.1111/j.2044-8295.1975.tb01454.x

[45] J.D. Zamfirescu-Pereira, Laryn Qi, Björn Hartmann, John DeNero, and Narges Norouzi. 2025. 61A Bot Report: AI Assistants in CS1 Save Students Homework Time and Reduce Demands on Staff. (Now What?). In *Proceedings of SIGCSE TS 2025*. ACM, 1309–1315. doi:10.1145/3641554.3701864

[46] Jin Zhao, Marta Knežević, and Tanja Käser. 2026. Evaluating Answer Leakage Robustness of LLM Tutors against Adversarial Student Attacks. In *Proceedings of ACL 2026*. 30588–30617. doi:10.18653/v1/2026.acl-long.1412