On the Helpfulness of a Zero-Shot Socratic Tutor

Kevin Gold[1] and Shuang Geng[2]

[1]Faculty of Computing and Data Sciences, Boston University, Boston, MA 02215, USA
[2]Digital Learning & Innovation, Boston University, Boston, MA 02215, USA
Emails: `klgold@bu.edu`, `sgeng@bu.edu`

Abstract. An AI tutor with a backend of GPT-4 was used as a homework assistant in an introductory data science course with 127 students. The tutor was given the homework assignments and the solutions, but was instructed not to give answers directly, and instead reply to student queries with questions meant to lead the student in the right direction, deploying a strategy known as the Socratic method. All interactions were anonymously logged and coded on seventeen attributes such as helpfulness and degree of answer leak. Surveys were also deployed to better understand students' perceptions and use cases. The tutor was generally perceived as helpful to learning by students and an independent coder, and only rarely directly leaked the solution. However, the system also displayed a pattern of student achievement observed in other tutoring systems, where students who asked too many questions did worse on the midterms, even controlling for programming background. Other trends in the usage of the tutor are also discussed.

Keywords: AI tutoring · Large language models (LLMs) · Generative AI · ChatGPT · Educational technology · Qualitative content analysis · Higher education · Programming education · Socratic method.

1 Introduction

The traditional practice of assigning homework almost certainly needs to be rethought in the age of large language models (LLMs), particularly in classes where publicly available LLMs like ChatGPT can provide passable answers to most introductory homework problems [11]. Rather than attempting to ban LLMs, a different approach is to offer an LLM that can provide a more educational experience while still furthering the student's immediate goal of progressing in the homework. The model can be a helpful teaching assistant, providing a student with hints and leading questions that get the student to the solution partly under their own power. Assuming students want to learn and, moreover, know they will be assessed on their learning later, this should prove a more attractive option than getting fully formed answers. But this plan faces a variety of questions at the outset: will students even want to interact with the custom LLM, instead of just using ChatGPT? Given that they do interact, will the LLM simply blurt answers? If the LLM withholds answers, will students find that unsurprising or frustrating? And the ultimate question: does such a system help the students learn?

This paper details the deployment of a GPT-4-based AI tutor in an introductory data science class of 127 students. By going to the AI tutor's website, students could select which homework they wanted help with, then ask the AI a question specific to that homework, such as, "I'm stuck on question 4c." The AI tutor, instructed to employ the Socratic method, would then respond with a question of its own, like, "What part are you having trouble with?" or "Are you familiar with how logical operators work in Python?" The exchange would continue with student queries and the AI's leading questions until the student left the interaction. The tutor had no "fine-tuning" nor examples of successful tutoring to do "few-shot" learning with; it was "zero-shot" to create a design as subject-matter-agnostic as possible. It simply had the instruction that it was to be a helpful TA who wanted the students to learn; the additional instruction to always reply with questions meant to lead the student to the right answer (generally referred to in education as the Socratic method); and it had the homework and answer key. Our fundamental question was: how far could a tutoring agent get using just GPT-4, the answer key, and the Socratic method?

All student interactions with the AI were logged, and later coded according to their helpfulness and other attributes to be described below. As an additional way to analyze the effectiveness of the AI tutor, students were surveyed at mid-semester and at the end of the semester to reflect on their experience with it. The survey at the end of the semester, in particular, was not anonymous, allowing us to connect survey responses to performance data in the form of midterm grades.

Our more specific research questions are: 1) How helpful was the AI tutor perceived to be? 2) How well did the AI tutor safeguard the answers? 3) Did the AI tutor effectively help the students learn the material, as measured by their performance on the in-class midterm exams?

Section 2 will explore some related work in this space. Section 3.1 will detail the design of the AI tutor. Section 3.2 will describe the class that the tutor assisted with. Section 3.3 will detail the system for coding the transcripts of the students' interactions. Section 3.4 will summarize the relevant parts of the survey. Section 4 will combine the results from our different sources. We then will offer some conclusions and next steps in Section 5.

2 Related Work

Evaluating what roles ChatGPT can play in instruction, previous work has rendered mixed results of varied performance across subject domains [7][11]. In [17], students in a random experimental group were given 10 days of access to ChatGPT in which they were encouraged to ask questions about the subject matter (vectors), and then reflect on the transcripts of their interactions. Students showed significant gains in performance as well as promising improvement on measures of self-regulation. [2] created LLM-powered agents for different roles, and while they did not measure the agents' effect on performance, they showed positive results in the areas of accuracy, fluency, empathy, engagement, and relevance. [10] examined the effectiveness of using ChatGPT to deliver worked-solution "hints," and found both a 30% error rate and performance that did not match that of a human tutor. [4] assessed the use of ChatGPT in writing tasks; the system seemed generally coherent and fluid, but had a clear bias toward positive sentiment over the instructor, and effects on student performance were again not assessed. In general, the effectiveness of LLM-driven tutors that generate hints for students has been heretofore inconclusive [11].

We also compare the results achieved with an LLM-based tutor to more traditional AI systems such as intelligent tutoring systems (ITSes) and computer-assisted instruction (CAI), or even to human tutors. Yet, the overall helpfulness of these other tutors has also been a matter of debate and may depend on context. For ITSes, which are historically expert systems, [6] reports an average gain in performance across studies of 0.66 standard deviations, with the caveat that most of the tests used to assess the students weren't standardized tests, but were instead "local" tests. [5] found that while an instructor who used an ITS as an opportunity to spend more time on personalized instruction saw student scores increase, no similar gain was found when instructors spent their time trying to get the system to work or doing something else. Older CAI systems seem to raise scores by about 0.3 standard deviations [15]. The effectiveness of human tutors has been a matter of some debate. [15] claimed that human tutors had an effect of 2 standard deviations on student performance, but later meta-analysis put the number closer to 0.4 [6]. LLM-based tutoring like ChatGPT introduces a novel dimension to the landscape, offering new affordances into the adaptive roles these mechanisms can play in educational settings. Despite the absence of specialized domain knowledge and pedagogical models inherent to traditional ITSes [6], the broad definition of ITSes enabling interactive learning environments [13] positions ChatGPT as an emerging educational tool with potentially similar effectiveness.

The Socratic approach (asking leading questions of the student) aims to address common challenges around student choices in interacting with tutoring systems. [14] found that while seeking help did not appear to improve a student's outcomes, accepting help (or doing learning tasks that a tutoring system suggested) did appear to improve student outcomes. [12] showed that avoiding help might sometimes be better than seeking it if the student was already unclear on the basic knowledge. [9] showed that students given access to nearly complete or complete solutions to their problems did worse than those who only received small hints. This set of evidence suggests that a question-driven tutoring approach that withholds solutions from students may lead to more effective learning for the students.

3 Methods

3.1 The AI Tutor

The AI tutor was hosted on the Hugging Face website. To begin an interaction with the AI, students would choose a homework number with a radio button, and then enter a query. The initial system instructions to GPT-4 were the following:

> You are a helpful teaching assistant in a data science course. Your primary goal is to help the students learn. This is the homework the student is talking about: [homework with solutions].

Then, every statement from the student was passed on to GPT-4 with the prefix:

> For this query, answer with a single question that you haven't asked before that is meant to lead someone in the right direction, without directly answering the relevant homework question - unless the problem is solved completely, in which case, quit.

The wording of the query prefix was designed to address two issues that came up in early tests: 1) the AI tutor sometimes repeated the same question over and over, and 2) the tutor did not know when to quit asking questions.

Each student's original query and the AI's response were appended to a transcript, and the whole transcript was displayed to the student. (By a policy of the academic unit, students were expected to submit transcripts of any interactions with AI along with their homework.)

An example transcript of a session with the AI tutor follows.

> Student: How do I go about solving problem 3 part b
> AI: You might want to consider what it means for time to be in non-military format. How does military time (24 hour clock) convert to non-military time (12 hour clock)? What would the time "three hours later" be in both formats?
> Student: what does it mean in question 3 part c when it says where X is the correct number of hours
> AI: The "X" in this context refers to the number of hours remaining until either noon or midnight, depending on the current time. So, if the current time is 10 AM, how many hours are left until noon? Or if the current time is 2 PM, how many hours are left until midnight?
> Student: for question 4, part f, where do I begin?
> AI: Think about how you would access elements in a regular list. Now, how might that change when you're dealing with a list of lists? How can you access the second element of the second list within the "prices" list?
> Student: for question 4, part f, what does it mean by Write the expression that accesses the second milk price? Is the problem asking to print the value or create a program that find the value
> AI: The problem is asking for an expression that retrieves the second milk price from the list. It's not asking you to print the value or create a specific program. So, how would you reference or access the second value in the milk prices list, given its position within the larger "prices" list?

3.2 The Course

The course that the AI tutor aided was an introduction to the data science major that focused on learning to code in Python for the first half, and assorted data science topics including machine learning, visualization, and asymptotic analysis for the second half. 127 students were enrolled.

The first midterm, delivered halfway through the course, concentrated on coding in Python, with multiple choice questions such as "What output is expected from the following code?" and "Which line should be corrected to make this code function?" in addition to one paper-and-pencil programming question. The second midterm, delivered four-fifths of the way through the course, contained review questions about programming, multiple choice questions about other data science topics (proper machine learning practices, visualization, etc), and another paper-and-pencil free response programming question. Both midterms were wholly in-class and "paper-and-pencil," and served as insurance that students could not get good grades by just relying heavily on generative AI; the hope was that this setup would more honestly assess students' learning and serve as an incentive throughout the course to not over-rely on tools such as the tutor.

The homework throughout the course was delivered via Jupyter notebooks, where students would usually write code to answer a prompt ("Write a function that ..."), but they might occasionally need to answer a thought question (such as needing to analyze the running time of some code, or explain that they are observing the phenomenon of overfitting). The tutor would thus mostly be explaining how to approach the programming problems, but it could also approach other problems, for example, encouraging students to think about how many nested loops are in the code when approaching an asymptotic analysis question.

There were 8 homework assignments in total. The topics covered include: introductory variables, branching, and other essentials; introduction to matplotlib, numpy, and loops; functions and dictionaries; dataframes and string manipulation; files and using statistical tests; object-oriented programming and recursion; machine learning with scikit-learn; and SQL, advanced pandas, and complexity. The lowest homework score was dropped, so not all students did the final assignment.

For help outside the tutor, one instructor, two graduate teaching assistants, and one undergraduate course assistant each offered three hours a week of office hours. The course met three times a week. The textbook, Deitel and Deitel's Introduction to Python for Computer Science and Data Science, was recommended to students as an additional source of practice problems, but its structure only loosely matched the structure of the course, and the book was optional.

Students completed final projects in teams where they analyzed the datasets of their choice using Python, and they would have had the perspective of having done this when completing our second survey about the tutor (see below).

3.3 Coding the Logs

All sessions with the AI tutor were assigned a unique random session number and anonymously logged. After the end of the course, a random sample of 20 sessions per homework, or 160 sessions in total for eight homework assignments, was taken and broken down by problem number, resulting in a total of 347 problem interactions recorded. Codes were assigned to complete interactions over individual homework problems instead of sessions since a single session could contain many interactions about different problems, each with its own degree of helpfulness, degree of answer leakage, and other attributes. For example, our aforementioned example transcript was coded into three problem interactions: problems 3B, 3C, and 4F.

An initial set of 25 sessions was coded by the authors (one being the instructor of the course) and an independent coder with a strong background in data science and Python to establish calibration. In this phase, a combination of deductive and inductive coding approaches was used to construct a set of initial interaction classifications. The classifications were selected to measure whether the tutor was fulfilling its intended educational purpose, and also to guide future improvements. The calibration set was not included in the final results, but used to achieve alignment on the codes. After the coders achieved alignment, the independent coder coded all problem interactions included in the sample for analysis, with regular check-ins with the authors to address points of uncertainty and ambiguity.

Our final codebook includes 17 dimensions under four parent categories: A) Helpfulness of Advice; B) Bot Failures: 1-Leak Correct Answer, 2-Clear Error in Answer, 3-Provide Irrelevant Answer, 4-Bot Demands Extra Work, 5-Fail to Point Back to Course Materials; C) Student Misuses: 1-Select Wrong Homework in GUI, 2-Spam for Hints, 3-Unclear Prompt, 4-Search for Exploits; and D) Student Question Types: 1-Debug Request, 2-Review Code, 3-Improve Style, 4-Clarify Concept, 5-Ask for Example, 6-Recommend Resource, and 7-General Hint Request. Each of the 347 problem interactions was assigned one value on each of the 17 dimensions. All dimensions are coded as Boolean variables (1 = yes, 0 = no) except A and B1, which are ordinal by the level of intensity.

The rubric for the ordinal ratings were as follows. For A-Helpfulness of Advice, a 3 (Very Helpful) indicated an answer that was as good as could be expected from a human teaching assistant. A 2 (Moderately Helpful) indicated a response that was not ideal, but did help the student somehow (suggesting a brute force approach for a logic problem, for example). A 1 indicated an unhelpful reply, and 0 was reserved for cases where the helpfulness couldn't reasonably be rated, such as an interaction with a student trying to break the system.

For B1-Leak Correct Answer, a 3 indicated a complete leak - the bot dumped the answer that it had been given. A 2 indicated a major leak - whatever the main idea of the problem was, the bot divulged it instead of guiding the student to it. A 1 indicated a minor leak - the bot revealed details of aspects of the problem, but not the main idea of the problem. A 0 indicated that the bot leaked no significant information about the problem.

3.4 Surveys

Two surveys were administered to better understand student views of the tutor. The first survey, the mid-semester survey, was part of a typical midterm survey, which asked for student feedback on a variety of resources, e.g., "[h]ow helpful has X been in learning the material?" Students could answer "didn't use" or reply on a 5-point scale from Very Unhelpful (1) to Very Helpful (5). The resources rated in this way included the lectures, the homework, the textbook, the office hours of each individual teaching assistant (TA) and course assistant (CA), ChatGPT, and the AI tutor. (The version of ChatGPT wasn't specified, but it was presumably the free 3.5 version for most students.) Free-response comments about these topics were also allowed. The survey was optional, anonymous, and not restricted to users of the AI tutor. It was administered after the fifth homework (out of eight), and after the first of the two midterm examinations. 39% (50 students) of the class completed the survey and 68% (34 students) of the respondents indicated that they used the AI tutor.

The second survey, the end-of-semester survey, was administered near the end of the semester, after the remaining three homework assignments and the second, cumulative midterm that covered all the homework. Like the first survey, it was optional, but offered a 5-dollar incentive to encourage participation. This resulted in an increase in participation: 50% (63 students) of the class completed the survey. Among the respondents, 71% (45 students) indicated use of the AI tutor. Because the students were not anonymous, we could link their answers to their performance, particularly midterm grades, since the in-class midterms were an indicator of how well the students had learned the material when separated from AI help. The survey asked the students: What was your level of programming experience entering the course (from 1=no experience to 4=very experienced)? Which assignments did you use the AI tutor on? Which of the following categories of questions did you ask the AI? And, what impact do you think the AI tutor had on your midterm grades?

Table 1 shows the demographic characteristics of students who voluntarily completed the end-of-semester survey (column header=Y), students who did not (column header=N), and all students enrolled in the course (column header=Overall). Chi-squared tests were performed to check for differences between the demographic compositions of respondents and non-respondents (except Fisher's exact test was used on IPEDS race/ethnicity status due to small values). The only statistically significant difference is seen on the non-resident alien indicator (Y=international students, N=domestic students, $p < 0.05$). While 83% of survey respondents are domestic students, only 66% of non-respondents are domestic, indicating that domestic students are more represented in the survey results. We see similar representation between male and female students and among students of different races/ethnicities.

Table 1. Demographic summary of DS110 students, broken down by whether the student has filled out the end-of-semester survey in column and by demographic characteristics in row.

		End Survey Respondent Indicator		
Characteristic	**Overall**	**Y**	**N**	**p-value**[2]
	$(N = 127)$[1]	$(N = 63)$[1]	$(N = 64)$[1]	
Gender				0.5
Female	62 (49%)	29 (46%)	33 (52%)	
Male	65 (51%)	34 (54%)	31 (48%)	
IPEDS Race/Ethnicity				0.4
African American or Black	10 (7.9%)	5 (7.9%)	5 (7.8%)	
Asian	33 (26%)	17 (27%)	16 (25%)	
Hispanic/Latino	8 (6.3%)	5 (7.9%)	3 (4.7%)	
Missing	5 (3.9%)	2 (3.2%)	3 (4.7%)	
Two or More Races Reported	4 (3.1%)	3 (4.8%)	1 (1.6%)	
U.S. Nonresident	33 (26%)	11 (17%)	22 (34%)	
White	34 (27%)	20 (32%)	14 (22%)	
Non Resident Alien Indicator				0.03
Y	33 (26%)	11 (17%)	22 (34%)	
N	94 (74%)	52 (83%)	42 (66%)	

[1] *n* (%)

[2] Pearson's Chi-squared test; Fisher's exact test

4 Results

RQ1. How helpful was the AI tutor perceived to be? Figure 1 shows that the AI tutor was generally perceived by students to be helpful relative to other resources they could use for help.

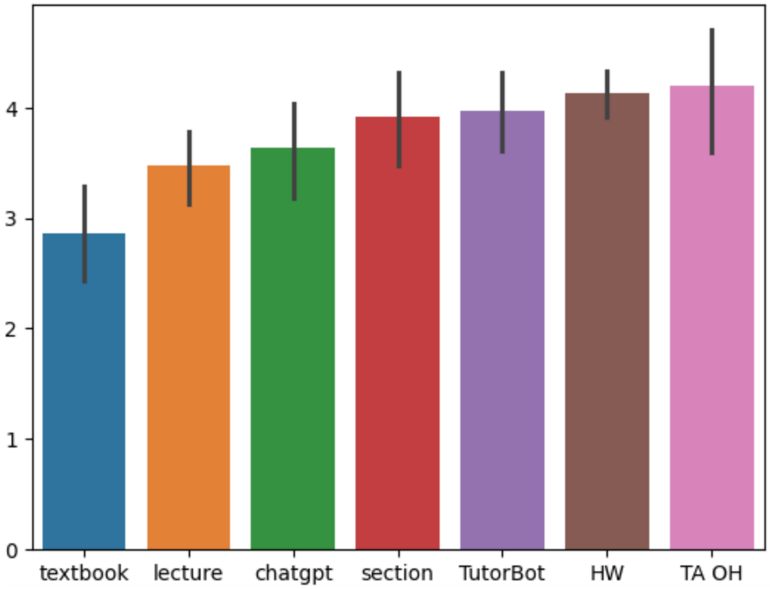

Fig. 1. Mid-semester student ratings of how well different resources "helped you learn the material" on a 5-point scale. Bars are 95% bootstrap confidence intervals.

The coder perceived the AI to offer mostly very helpful advice both before and after the first midterm. Before the first midterm, 71% of the logs (related to assignment 1 to assignment 6) were rated as maximally ("Very") helpful, 23% as moderately helpful, and 6% as not helpful. This pattern remains similar after the first midterm, where 74% of the logs from the two assignments after the first midterm were deemed very helpful, 13% were deemed moderately helpful, and 13% were deemed not helpful. A Chi-squared test conducted on helpfulness levels between the two periods resulted in a p-value of 0.01, suggesting that with harder problems that were more prevalent in later assignments, the AI tutor's advice was less likely to be moderately helpful and more likely to be unhelpful. Nevertheless, the AI was still perceived to be mostly helpful overall.

The students were more tempered in their assessment of the AI, but still generally favorable. Among the 50 respondents in the mid-semester survey, 34 reported having used it at least once. Their average rating on a 5-point scale from "Very unhelpful" to "Very helpful" was 4.0, or "Somewhat helpful." 29% of those who reported using the AI tutor indicated it was "Very helpful," 47%

indicated it was “somewhat helpful,” 15% indicated it was neither helpful nor unhelpful, and just 9% said it was some degree of unhelpful.

To put these results in perspective, the same survey also asked students to rate the textbook, lecture, ChatGPT, their discussion section, homework, and TA office hours, all on how much they helped the students learn. The average ratings for each of these were 2.9 for the textbook, 3.5 for the lecture, 3.6 for ChatGPT, 3.9 for the discussion sections, 4.1 for the homework, and 4.2 for each TA (the TAs happened to all have the same average). The sample sizes were small, though, for these calculations: just 30% of respondents reported using the textbook, 60% reported using ChatGPT, and the percent of students visiting each TA was 24%, 16%, and 12%. Still, using a 2×2 Chi-squared test that simply checked for a difference in frequency, we could show that the human TAs did get a rating of “Very Helpful” significantly more than the AI tutor ($p < 0.005$), but the AI in turn had a rating of “Very Helpful” significantly more often than the textbook ($p < 0.05$). The failure to show significance for the advantage over ChatGPT is perhaps disappointing, but considering that ChatGPT has no safeguard against simply solving the problem for the student, the Socratic tutor’s parity and non-significant advantage in mean helpfulness is still interesting.

The results of the survey at the end of the semester, after the students had completed all the homework, two midterms, and a final project, were still more positive. Of 45 respondents who indicated they used the AI tutor, 42% rated it as Very Helpful, 49% rated it as Moderately Helpful, and just 9% rated it as Not Helpful (on a 3-point scale consistent with the chatlog coding). This is more tempered than our coder’s view, but, responses that seem helpful to an expert may not seem so to a novice. Asked specifically about the impact on their midterm performance, 24% indicated their midterm performance was better or much better because of the AI tutor, 65% indicated it didn’t matter, and 11% indicated it was worse as a result.

The end-of-semester survey also asked students exactly what applications of the tutor they found helpful. Table 2 summarizes these responses. The tutor seemed to perform best when its answers were somewhat low-level, like describing Python syntax or offering suggestions about keywords and constructs to use. Clarifying the intent of the questions was also rated highly. The AI tutor was still rated as moderately helpful when performing more high-level tasks like debugging the student’s code or describing broadly how to approach problems. The last broad category of tasks were those where the students mostly didn’t think to ask, such as asking for references or practice problems. Even among these, when restricting the ratings to students who had tried the tasks, the only task where the most common rating was “not helpful” was getting a study guide for the midterm, which makes sense because the tutor had no knowledge of what material was fair game.

RQ2. How well did the AI tutor safeguard the answers? Even though we asked students in the surveys whether the AI tutor was helpful for their learning, we might think that they ignored some of the instructions and simply

Table 2. Ratings of tutor helpfulness regarding various tasks by students who indicated use of the tutor (N=45). Rating levels (last column) were Very Helpful, Moderately Helpful, Not Helpful, or Never Did That.

Use Task	Helpfulness Mode	Very/Mod/ Not/Never
Clarify what a question is asking	VERY	17/16/3/9
Ask about Python syntax	VERY	22/15/0/8
Ask about keywords or programming approaches	VERY	18/15/3/9
Confirm solutions are correct	VERY	14/12/8/11
Get help debugging	MODERATE	16/20/3/6
Get an example of how to do something	MODERATE	7/19/7/12
Ask how to approach problem	MODERATE	18/22/1/4
Get a reference to an external resource	NEVER	4/10/5/26
Get practice programming problems	NEVER	3/11/4/27
Get a study guide for the midterm	NEVER	1/8/10/26

thought it was helpful because it divulged answers. Keeping the answers from the students was also a major motivation for the AI tutor in the first place - it was intended to be an alternative to ChatGPT that did not immediately solve the problem for the student.

Figure 2 shows the average leak level by homework for the samples that were rated. The average leak level across the 8 homework assignments was 0.86. This is not bad, as a 1 indicated a minor leak that did not give away the essential point of the problem. Out of 347 total rated problems, 67 (19%) were rated a "major" leak (2), and 42 (12%) were rated a "complete" leak (3). Overall, leaks did not seem to be the source of students' positive attitudes toward the AI tutor.

RQ3. Did the AI tutor effectively help the students learn the material, as measured by their performance on the in-class midterm exams? This is a difficult question to answer. Students who need help are more likely to seek help, and students who need no help are unlikely to use the AI tutor. The degree of needing help is therefore a confounder for a straightforward comparison of scores. We also can't tie individual logs to midterm performance for this particular semester, because the logs were anonymous. We do, however, have the students' non-anonymous responses in the end-of-semester survey, and we can tie those to their grades in the course.

The survey recorded the degree of background in programming, ranging from no experience (level=0) to knowing most of the Python covered in the course already (level=3). We performed a Chi-squared test with this feature versus three levels of self-reported use of the AI tutor, ranging from usually wanting just a quick answer (level=1) to asking for complete walkthroughs of problems (level=3). A Chi-squared test showed that there was indeed a significant association ($p < 0.02$) between background level of experience and usage pattern; for

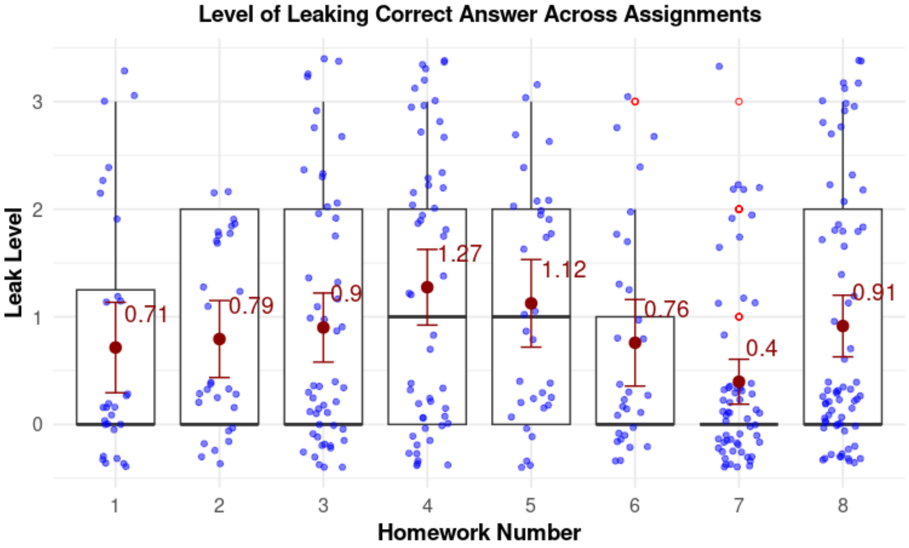

Fig. 2. Leak levels by homework assignment, where 1=minor leak, 2=major leak, 3=complete leak. Scores are jittered for readability. Red circles indicate outliers.

example, those with the most programming experience only ever reported using the AI tutor for quick checks.

On a hundred-point scale, the mean uncurved scores for the midterms for the groups reporting light, moderate, or heavy use were 75%, 73%, and 67%, respectively. But these raw scores clearly included the confounding effect we described earlier: more help-needing students asking for more help.

We attempted to control for programming background by performing subclassification [3]: treating the different levels of question asking as different treatments, finding the effect of each treatment on each subgroup of programming background, then finding the weighted average for each treatment based on the frequencies of the backgrounds. This resulted in the finding that light use of the tutor would result in a 6.25% drop in score, moderate use of the tutor would result in a 7.07% drop in score, and heavy use (indicating that the most common use case was “A walkthrough of the problem from beginning to end”) resulted in a 13.36% drop in score.

However, our results on other help sources that students reported cast doubt on whether this analysis has taken all confounders into account. Using similar calculations, but treating other sources of help as the treatments, we found that human tutor office hours net a mere 0.71% increase in midterm score, and the textbook hurts performance by 8.3%. ChatGPT, meanwhile, looks very good by this method, with an average gain of 4.3%, better even than human tutoring. These results should look suspicious, and they could be confounded by the students’ freedom to choose their methods of help, which we did not and cannot control for (short of randomly forbidding some students to use some resources). The kind of student who decided to use ChatGPT might be someone who feels very comfortable with technology, while the student who chose to work out of an optional physical textbook might be less comfortable. The student who chose to physically go to office hours was probably having problems that couldn't be solved easily by the other options, offsetting the gain from human help. And so on. (For a discussion of the importance of controlling for confounders but the difficulty in finding all of them, see [16].)

Another possibility for the gap between ChatGPT users and AI tutor users is that the students tended to use the tutor in rather limited ways compared to ChatGPT; they may not have fully realized the array of questions that they could ask. Out of all problem transcripts, just 68 (20%) included a question about a concept. Perhaps students thought of ChatGPT as a better source of conceptual explanations, given it did not reply to these requests with questions of its own.

Figure 3 shows how the frequencies of some important student behaviors changed over time from assignments 1 to 8. Fisher's exact tests with a 0.05 significant level are performed to compare frequencies before and after the first midterm (threshold denoted by the dotted line). Unclear prompts and general requests for hints did significantly decrease ($p < 0.05$), while the frequency of "code review" or checking answers significantly increased ($p < 0.0001$), climbing from 31% of all queries to 62%. The frequency of debug requests did not undergo significant change after the first midterm ($p = 0.054$).

5 Discussion

There may seem to be a paradox in these results at first - how can an AI tutor be widely regarded as helpful, by students and the log coders alike, but not show clear gains in student performance on the tests? If it were the promise of free solutions that made it seem helpful, why is the average "leak level" closer to minor than major? And why would the TA ratings be higher than the AI tutor's, if the student ratings were driven solely by the promise of free answers?

In all likelihood, we are still seeing selection effects here, despite the fact that we tried to control for student background. Whether students are new to Python or experienced, we could hypothesize that the "help-seeking" students are overall less likely to "tough it out" and make it to a solution by themselves. This could have effects on their scores even beyond the programming part of the midterm.

This bias was reflected in conversations with students when the AI tutor was suggested as a source of help. "No," a few students said, "I'd rather learn the material on my own." It has also been observed in the literature on intelligent tutoring systems that "contrary to many help-seeking theories, avoiding help (and failing repeatedly) is associated with better learning than seeking help on steps for which students have low prior knowledge" [12].

We also believe that this bias affected our result of nearly no net gain in seeking human office hours help. Surely, the TAs do help students learn during

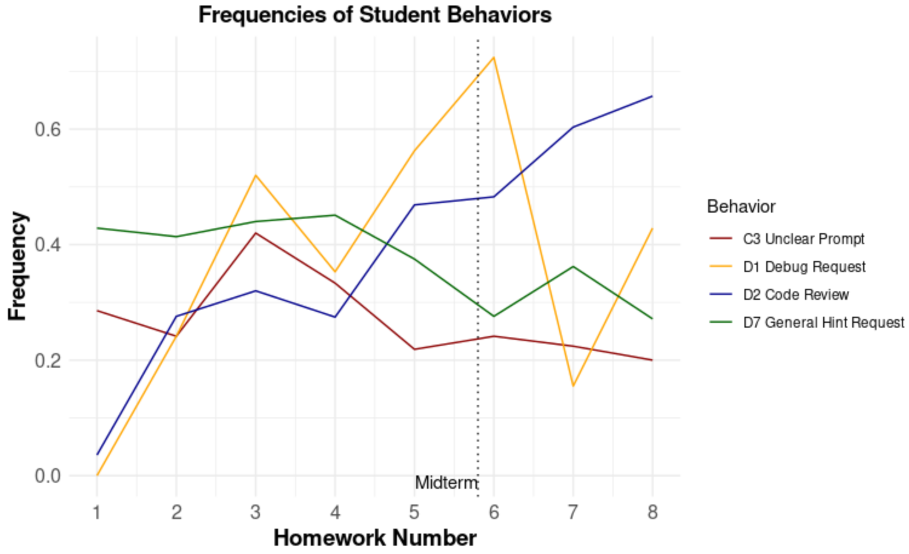

Fig. 3. Change over time in frequencies of four kinds of queries, three of which significantly changed from before the first midterm to after it (debug requests did not). Values may not sum to 1 because an exchange for a single problem could fall into multiple categories.

office hours, especially TAs who are seen as dedicated by the instructor and rated relatively highly (4.2/5) by the students. However, this is balanced out by the fact that the students who make the trek to the hours are often those who think they need quite a bit of help, rather than just a little; and they have decided to get help rather than wrestling with the problems alone. If we apply some kind of adjustment to account for this bias with the human tutor, then the AI tutor surely deserves some mental adjustment as well.

What was it that the students found so helpful about the AI tutor? At its best, the AI tutor would explain concepts that were related to the homework, even if not asked about them directly. For example, on a problem that used floating point numbers, a student got a result that showed floating point inaccuracy at several places after the decimal point, and the student asked whether their answer was incorrect. The AI said that this was a floating point error, and asked if the student wanted an explanation of it. The student said yes, and the AI delivered the explanation. Not only did the student learn about floating point error, but on reading the logs, the instructor realized that this topic wasn't explained thoroughly before the assignment went out, and the curriculum could be adjusted the next time. This embodied the ideal interaction with the AI tutor.

But also, commonly, the AI tutor was used to either debug problems with programs that weren't working (37% of problems) or check answers that seemed to be working (43%). In the former case, the AI will generally point out where the bug is, even if the fix is not presented, and in the latter case, the AI will generally assure the student that a correct answer is correct. These are both actions that could be perceived as helpful and educational, but they might not be in the long run. For debugging, it has been argued that the process helps students build a mental model of how the program executes [8], so circumventing this process with the help of the tutor may leave the students worse off. In the latter case, one of the authors has observed that students asking "is this right?" during office hours often have no confidence in their answer and need to be walked back to a point where they are comfortable, solving the problem all over again from there. In a rush to congratulate students - a pleasant action that GPT-4's reinforcement learning probably encourages - the AI tutor misses a pedagogical opportunity.

These issues aside, there is great promise for LLM-based tutoring in the very near future, and the challenge is to make genuinely educational LLM experiences more attractive to students than those that simply provide a quick fix. This work shows that a wealth of value is available "out-of-the-box" with zero-shot GPT-4 tutors when merely prompted with assignment solutions alongside a directive to be helpful, use questioning techniques, and refrain from providing direct answers. Notably, the tutor embodies succinctness, encouragement, and a propensity to offer hints rather than explicit explanations in its responses — none of which were directly requested in the initial instructions, but all of which are beneficial for student learning. The findings herein highlight the significant potential of this approach to be effectively and easily adapted across various other disciplines outside data science.

What might it look like to port this tutor to different disciplines? For disciplines with quantitative homework questions, it would seem the tutor has a good chance of being ported with virtually no changes - just a different homework to ingest. The design's incorporation of the actual solutions makes it difficult for the bot to go too far astray; at worst, it may supply an erroneous reason for a correct approach (e.g. "ignore friction here because the problem involves circles"). Educators looking to try out the approach might first try presenting various problems with solutions to GPT-4 and ask it to justify the answers. We would expect justifying answers to be easier than producing them, and so we'd generally expect the system to work reasonably well across quantitative disciplines. What may work less well is applying the system to homework in disciplines that involve writing - there, leading students to "solutions in hand" may stifle students in coming up with their own takes on the subject matter, and a better use of AI may just be as automated essay critiquer on axes such as clarity and originality.

Another place for systems like this one to take hold could be massive online courses. Face time with course staff is at a premium in said courses, and automated feedback is much better than no feedback. The main way in which the system does not currently scale up is the use of a single API key, which runs in to rate-limiting problems. But the hurdles to using multiple API keys and load balancing are understood issues that shouldn't be insurmountable.

There still remains work in the next iteration to further refine the AI tutor, mostly on three dimensions. First, we aim to incorporate some elements of few-shot learning, where the AI tutor examines its own tentative transcript before releasing it to the student user. The additional step will take the entire interaction into consideration, and interrupt the interaction when needed with a request to encourage students to work on the problem independently. Secondly, we plan to introduce a feature that allows students to specify when in the course they last felt comfortable with the material, allowing the tutor to tailor the interaction to their proficiency. Lastly, we plan to augment the tutor to generate additional practice problems similar to those in the assignments. Shifting to a model of more low-stakes work could help students get more valuable practice and feedback. But when an LLM like ChatGPT offers complete solutions to problems for free, encouraging students to engage in more work is a delicate balancing act.

References

1. Baker, R., Corbett, A., Koedinger, K., Wagner, A.: Off-task behavior in the cognitive tutor classroom: when students "game the system." In: Proceedings of CHI 2004: Computer-Human Interaction, pp. 383—390. ACM, New York (2004)
2. Cao, C., Ding, Z., Lin, J., Hopfgartner, F.: AI chatbots as multi-role pedagogical agents: Transforming engagement in CS education. arXiv:2308.03992 (2023). https://arxiv.org/abs/2308.03992
3. Cunningham, S.: Causal Inference: The Mixtape. Yale UP, New Haven and London (2021). https://mixtape.scunning.com/
4. Dai, W., Lin, J., Jin, F., Li, T., Tsai, Y., Gasevic, D., Chen, G.: Can Large Language Models Provide Feedback to Students? A Case Study on ChatGPT. (2023). https://doi.org/10.35542/osf.io/hcgzj
5. Koedinger, K.R., Anderson, J.R.: Effective use of intelligent software in high school math classrooms. Artificial intelligence in education: Proceedings of the world conference on AI in education, pp. 241–248. Association for the Advancement of Computing in Education, Charlottesville, VA (1993)
6. Kulik, J. A., Fletcher, J. D.: Effectiveness of Intelligent Tutoring Systems: A Meta-Analytic Review. Review of Educational Research, **86**(1), pp. 42–78 (2016). https://doi.org/10.3102/0034654315581420
7. Lo, C.K.: What Is the Impact of ChatGPT on Education? A Rapid Review of the Literature. Educ. Sci., **13**, 410 (2023). https://doi.org/10.3390/educsci13040410
8. Lowe T.: Debugging: The key to unlocking the mind of a novice programmer? In: 2019 IEEE Frontiers in Education Conference (FIE), pp. 1–9. IEEE, New York (2019)
9. Mathews, M., Mitrović, T., Thomson, D.: Analysing High-Level Help-Seeking Behaviour in ITSs. In: Nejdl, W., Kay, J., Pu, P., Herder, E. (eds) Adaptive Hypermedia and Adaptive Web-Based Systems. Lecture Notes in Computer Science, vol. 5149. Springer, Berlin, Heidelberg. (2008). https://doi.org/10.1007/978-3-540-70987-9_42
10. Pardos, Z., Bhandari, S.: Learning gain differences between ChatGPT and human tutor generated algebra hints. arXiv: 2302.06871 (2023). https://doi.org/10.48550/arXiv.2302.06871

11. Prather, J., Denny, P., Leinonen, J., Becker, B., Albluwi, I., Craig, M., Keuning, H., Kiesler, N., Kohn, T., Luxton-Reilly, A., MacNeil, S., Petersen, A., Pettit, R., Reeves, B., Savelka, J.. The Robots are Here: Navigating the Generative AI Revolution in Computing Education. In: Proceedings of ITiCSE 2023. https://arxiv.org/abs/2310.00658
12. Roll, I., Baker, R., Aleven, V., Koedinger, K.R.: On the benefits of seeking (and avoiding) help in online problem-solving environments. J. Learn. Sci. **23**(4), pp. 537–560 (2014). https://doi.org/10.1080/10508406.2014.883977
13. Steenbergen-Hu, S., Cooper, H.: A meta-analysis of the effectiveness of intelligent tutoring systems on college students' academic learning. Journal of Educational Psychology, **106**(2), pp. 331–347 (2014). https://doi.org/10.1037/a0034752
14. van Stee, E., Heath, T., Baker, R., Andres, J.M.A., Ocumpaugh, J.: Help seekers vs. help accepters: Understanding student engagement with a mentor agent. In: Proceedings of AIED 2023. pp. 139–150 (2023)
15. VanLehn, K.: The behavior of tutoring systems. International Journal of Artificial Intelligence in Education, **16**, pp. 227–265 (2006)
16. Weidlich, J., Gašević, D., Drachsler, H.: Causal inference and bias in learning analytics: A primer on pitfalls using directed acyclic graphs. Journal of Learning Analytics **9**(3), pp. 183–199 (2022). https://doi.org/10.18608/jla.2022.7577
17. Wu, T.-T., Lee, H.-Y., Li, P.-H., Huang, C.-N., and Huang, Y.-M.: Promoting Self-Regulation Progress and Knowledge Construction in Blended Learning via ChatGPT-Based Learning Aid. Journal of Educational Computing Research, **61**(8), pp. 3–31 (2023). https://doi.org/10.1177/07356331231191125