The Education Column

BY

Dennis Komm and Thomas Zeume

ETH Zurich, Switzerland and Ruhr University Bochum, Germany

dennis.komm@inf.ethz.ch and thomas.zeume@rub.de

Good In Theory

How Do LLMs Perform In a Theory Course?

Moritz Stocker
Department of Computer Science
ETH Zurich
moritz.stocker@inf.ethz.ch

Dennis Komm
Department of Computer Science
ETH Zurich
dennis.komm@inf.ethz.ch

Abstract

We study the performance of two large language models in an undergraduate second-year Theory of Computing course. The course contains two pen-and-paper exams that are each taken by around 450 students. We had both exams solved by two recent versions of GPT models, with the goal of comparing their scores to those of the students. To avoid any possible bias, we copied the solutions manually, and added them to the stack of exams that were graded by the unsuspecting teaching staff. Our results show that GPT-4o would not have passed either exam, while GPT-5 would not only have passed, but would actually have scored better than the majority of students.

1 Introduction

Following the introduction of ChatGPT to the general public in November 2022, one of the most pressing implications discussed early on was the potential impact of large language models (LLMs) on education [9]. This impact may come in many forms, e.g., with respect to utilizing LLMs and the like in classrooms or lecture halls, or diminishing (or at least re-evaluating) the meaningfulness of homework or take-home assignments. The debate is ongoing, with some researchers pointing out great opportunities for LLM-assisted teaching [13], while other voices are more skeptical [11, 12].

Whatever the concrete use case, one central underlying question is what LLMs are actually capable of doing, i.e., how well they perform in solving certain tasks. In this paper, we focus on undergraduate CS, and more specifically on the performance of two GPT-based models when solving exams of a second-year university course on the *Theory of Computing*. The course content is somewhat standard [8] and comprises *formal languages*, *computability* and *complexity theory*, with around 450 students attending each year.

LLMs are increasingly successful when solving problem statements from different domains—including CS—, and there is a rich body of literature on benchmarking different models, e.g., when solving problems from specific databases.

We follow a different approach. In particular, our goal was to assess LLM performance in a setting that is as close as possible to solving an exam under real-world conditions:

1. *Genuine exam.* We had two models (GPT-4o and GPT-5) work on two real exams that were part of the aforementioned undergraduate course.

2. *Blind grading.* The solutions were graded by teaching staff, who were unaware that two solutions (among the roughly 450) in each exam had not been created by humans.

3. *Lower bound.* We purposely did not train the models on old exams or the course material. We intentionally used two "vanilla" GPT versions without any kind of fine-tuning to determine what is possible in such a *zero-shot* setting.

One motivation was to investigate what an unprepared student having access to a generic LLM for a short period of time—say, during a bathroom break—could gain in an exam.

Concretely, we posed the following two research questions:

- **RQ1.** How well do LLMs, without any fine-tuning or special training, perform in *Theory of Computing* exams?

- **RQ2.** Are there specific types of tasks where LLMs outperform humans and vice versa?

Our results show that, unsurprisingly, GPT-5 consistently outperformed GPT-4o; and while the latter is "below the bar" in both exams, the former would actually receive roughly 80–90% of the points in each of them. Taking into account that some of the points were deducted due to GPT-5 using a notation different from that introduced in the course (without properly introducing it as part of its solution), this is indeed an impressive result.

2 Related Work

As mentioned above, the potential applications of LLMs in educational settings are numerous and subject to ongoing research. A very recent survey on LLMs in education is given by Shi et al. [20], drawing from 88 studies that consider different applications in different settings. Our research questions focus on pure performance in a very specific domain—and under conditions that are as close to a real-world setting as possible.

A great deal of early work on benchmarking LLM performance in education was—not very surprisingly—focused on programming courses and competitions. Pereira and Mello [18] provide a recent overview on how LLMs can assist educators in programming classes, e.g., with generating feedback, debugging code, or personalizing learning. The performance of tools like *GitHub Copilot* received considerable attention in the early days of the "GenAI hype" due to their abilities in solving, say, almost half of the problems stated in a CS1 course on the first try, as reported by Deny et al. [5]. Since then, programming benchmarking has shifted from solving simple tasks, e.g., from the *Mostly Basic Python Problems Dataset* [1] to more challenging problem statements, e.g., from platforms such as *LeetCode* [16]. *AetherCode* [24] is a particularly challenging benchmark, sourcing problems from prestigious competitive programming competitions, such as the *Olympiad in Informatics* and the *International Collegiate Programming Contest*.

While we also applied LLMs within the domain of CS, we considered problem statements from *Theory of Computing*, which lies at the intersection of CS and discrete mathematics. By now, there is a rich literature on the performance of LLMs on different math problems. These works often use established datasets such as *GSM8K* [4] or *MATH* [15]. Similar to programming benchmarks, one source of problem statements are math (and physics) olympiads, such as with *OlympiadBench* [14] or *OlymMATH* [22]. *Math Arena* evaluates the performance of different models "on the latest math competitions and olympiads" [2]. Zhang et al.[1] introduced "a curated dataset of 4,550 questions and their solutions spanning exams and assignments from all courses that form the curriculum for MIT's Mathematics and Electrical Engineering and Computer Science majors" [25]. Employing sophisticated prompt engineering, GPT-4 remarkably achieved "a perfect solve rate" [25] working on a test set (not containing image-based problem statements).

Our goal was to directly compare the performance of LLMs to that of humans, which is somewhat in the spirit of the original benchmark of all AI: the seminal *Turing test* [23]. As mentioned above, the results presented in this article were obtained under conditions that are as authentic as possible. In particular, GPT-

[1]The most recent version of the paper is withdrawn due to not receiving "permission to release the data or model fine-tuned on the data."

generated solutions were graded by TAs who were unaware of this fact—and who had no reason to believe that LLMs were involved in any way.

A similar approach was followed by Richards et al. [19], who received passing scores with GPT-generated solutions to undergraduate end-of-module assessments of different CS modules; the LLM performed much worse on postgraduate material. Staff graded the solutions as part of a "quality assurance" marking exercise and were given five GPT-generated solutions alongside ten student-generated ones. As in our setting, the markers were not aware that artificially generated solutions would be included in their workload. However, due to the much higher ratio of artificial solutions to student solutions, combined with the fact that the exams were solved remotely, the graders collectively flagged all artificial solutions as suspicious. In our case, no such suspicions were raised.

The setting explored by Ding et al. [6] is also rather close to ours; in their work, take-home assignments from an undergraduate *Algorithms* course were solved by two GPT-based models (GPT-4o and o1-preview), and these solutions were then blindly graded by the teaching staff. Problems required combinatorial proofs, such as computing the number of distinct spanning trees in a given complete graph. Their findings include that o1-preview consistently outperformed GPT-4o, and while the latter did not receive a passing score, the former did—even obtaining 92% of the points. Moreover, GPT-4o provided many more unjustified / misleading claims in its reasoning. One key difference from our work is that they used take-home assignments while we use on-campus exams.

Also related to our work is the research by Li et al. [17], which investigates how well then-current GPT models were able to solve exam-level CS tasks. Their set of questions was broader, but did include a number of problem statements regarding *Theory of Computing*. They examined in great detail the response quality of the models when provided with increasingly advanced hints. Their research does not include, however, a direct comparison to student results.

The work by Golesteanu et al. [10] and Dougherty et al. [7] is perhaps the most similar to our approach. The authors also analyzed the scores and error types of a GPT model [10], and compared the scores between two models, one of which was more advanced [7]. One major difference to our approach is that the points were assigned with the knowledge that the answers were provided by an LLM.

Chang et al. [3] provide an overview of how LLMs have been evaluated; their survey covers many more applications than education.

3 Methodology

Our lab teaches an undergraduate *Theory of Computing* course at ETH Zurich. This course is mandatory for second-year CS students and a popular elective for

second-year math students. It is taken by around 500 students each year. As part of this course, students may take a midterm and an end-of-term (final) exam;[2] both are pen-and-paper exams, manually graded by the teaching assistants (TAs) of the course. Each exam lasts 90 minutes, is taken on campus (supervised by the lecturers and TAs), and no auxiliary materials are permitted.

In the 2025 fall semester, we used two GPT-based models (*Azure GPT-4o* and *Azure GPT-5*) to complete two copies of these exams, both for the midterm and the final. We proceeded in three steps:

- *Step 1.* The tasks were supplied to the respective LLM exactly as stated on the exam, without providing the models with lecture notes, old exams, or sample solutions.

- *Step 2.* We then manually copied (i.e., wrote by hand) the answers onto regular exam paper and added these copies to the exams completed by the students for grading. The only change made to the model-provided answers was to convert the *deterministic finite automata*, which the LLMs gave as tables, into the expected graphical form. Due to standard procedure, all exams, including these copies, were anonymized before the TAs received them.

- *Step 3.* The TAs were not informed that LLM-generated responses were included among the exams and graded them as part of their regular duties. Each question was separately graded by a group of TAs; ensuring that each TA consistently graded the same task across all exams. The authors were not involved in the grading process of the LLM-generated solutions.

After the grading of the final was completed, the TAs were informed about the experiment and offered the opportunity to receive detailed information.

IRB approval was granted by the ethics board of ETH Zurich and can be provided on request.

3.1 Exam Questions

The problem statements in the two exams could be roughly divided into three categories: MULTIPLE-CHOICE, RECALL, and TRANSFER. In the MULTIPLE-CHOICE category (questions 2 and 6 in the midterm, questions 2 and 3a in the final), students received

[2]The grading scheme in use grades on a scale from 1 to 6 in steps of 0.25, with 4 being the first passing grade. In our course, the final grade of a student is the average of the grades of the midterm and final, rounded to a multiple of 0.25. There is a peculiarity: a student can discard the grade and instead take a so-called "session exam" that takes place after the end of the course and then gives the final grade.

1 point for each correct answer, no points for an incorrect answer, and 0.5 points if the question was left unanswered. As an example, consider question 2a in the midterm:

Midterm: Question 2a

For each of the following languages, decide whether it is regular or not.

1. $L_1 := \{w \in \{a,b\}^* \mid w = xx^{\mathsf{R}} \text{ for some } x \in \{a,b\}^*\}$
2. $L_2 := \{x \in \{0,1,2,3,4,5,6,7,8,9\}^* \mid x \text{ is the decimal representation of a number divisible by } 17\}$
3. $L_3 := \{a^n w \mid w \in \{a,b\}^*, n \in \mathbb{N} \text{ and } |w|_b = n\}$
4. $L_4 := \{a^n w \mid w \in \{a,b\}^*, n \in \mathbb{N} \setminus \{0\} \text{ and } |w|_b \geq n\}$

In the RECALL category, students were essentially asked to reproduce, in one way or another, information that they had seen explicitly in the lecture (question 5 in the midterm, questions 3b, 3c, and 4 in the final). As an example, consider question 3c in the final:

Final: Question 3c

Order the following complexity classes by their known inclusions: P, DLOG, EXPTIME, NPSPACE, NLOG, NP.

The TRANSFER category contained exercises where students had to apply techniques from the lecture to problems they had not encountered yet (questions 1, 3, and 4 in the midterm, questions 1, 5, and 6 in the final). As an example, consider question 4 in the midterm:

Midterm: Question 4

Show that any deterministic finite automaton that recognizes the language

$$L := \{a^n bw \mid n \in \mathbb{N}, w \in \{a,b\}^*, \ w \text{ contains neither the subword } aa \text{ nor the subword } bb\ \}$$

over the alphabet $\Sigma = \{a,b\}$ must have at least 5 states.

4 Results

Of the two models, GPT-5 was unsurprisingly more successful in both exams. Its answers received very high scores, though not resulting in a best grade, while the

Table 1: Number of students compared to which the GPT-models scored worse, equal to, or better in the two exams, as well as the final grade. The second row contains only the students that also participated in the final, allowing for a better comparison between the two exams. The same holds for the final grade. Note that the final grade is less fine-grained than individual points.

	GPT-4o			GPT-5		
	better	equal	worse	better	equal	worse
midterm	6	3	464	367	28	78
midterm*	0	2	427	324	28	77
final exam	64	7	358	335	9	85
final grade	5	10	414	345	51	33

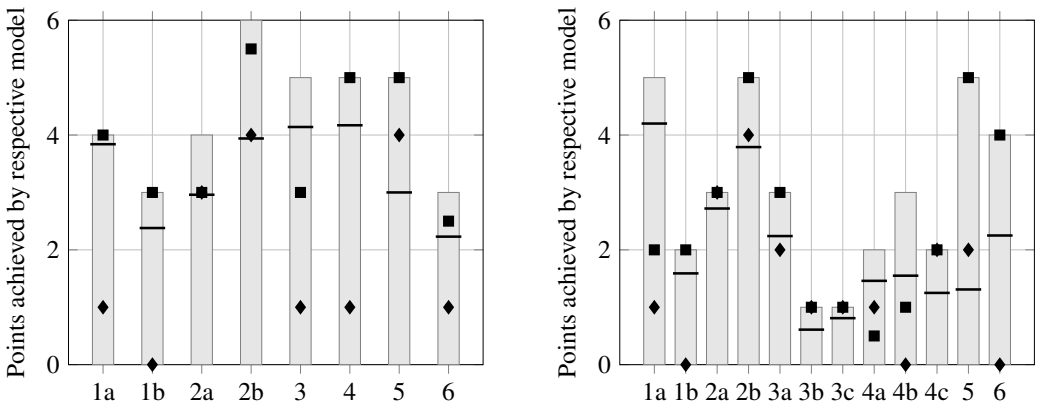
answers by GPT-4o did not receive a passing grade in either exam.

- In the midterm, 473 students participated. GPT-5 reached 31 out of 35 points, receiving more points than 78% of the students. GPT-4o received 15 points, beating only 1% of student scores.
- In the final, 429 students participated. GPT-5 reached 29.5 out of 36 points, again receiving more points than 78% of students. GPT-4o received 17 points, beating 15% of students.
- Over both exams, GPT-5 reached a grade of 5.5, beating 80% of students. GPT-4o received a grade of 3.25, once again beating only 1% of students.

The concrete numbers are shown in detail in Table 1. Comparing the relative performance of the two models in both exams is complicated by the fact that the students who chose not to take the final after participating in the midterm were usually those with weaker scores. If we only consider the students that also took part in the final, GPT-5 beat only 76% of student scores in the midterm, while no students performed worse than GPT-4o (row *midterm** in Table 1). Thus both models scored slightly better in the final than the midterm relative to the students. The individual points are shown in Figure 1.

4.1 Errors Made by Azure GPT-5

The more successful model GPT-5 reached 31 out of 35 points in the midterm, and 29.5 out of 36 points in the final. The points lost appear to fall into four broad categories. Note that these error categories are independent of the question categories described in Section 3.1.



(a) The six tasks of the midterm

(b) The six tasks of the final exam

Figure 1: The performance of both models for both exams. The gray bars indicate how many points were obtainable for the individual tasks. The horizontal line indicates the average number of points achieved by students; ♦: GPT-4o, ■: GPT-5

Background Knowledge. For many concepts covered in a *Theory of Computing* course, there are conflicting definitions. Usually, they are equivalent for all intents and purposes. However, to teach these concepts and argue about them, it is important to choose one such definition and stick with it.

In our course, we sometimes chose a definition that is not the one usually found in the literature. Since the GPT model was given the problems without direct access to the lecture material, its answers were unsurprisingly occasionally based on other definitions. As an example, consider questions 4a and 4b in the final:

Final: Question 4

Let $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{\text{accept}}, q_{\text{reject}})$ be a 1-tape-TM that always halts with a space complexity of $s(n)$ for some function s with $s(n) \geq \log_2 n$.

(a) Describe the 4 components constituting an inner configuration of M.

(b) Show that there is a constant c such that the number $|\text{InConf}(n)|$ of possible inner configurations of M on instances of length n can be bounded from above by $c^{s(n)}$.

These two questions were meant to check knowledge of a specific proof technique shown in the course. In the course, a 1-tape-TM was defined as a multi-tape-TM (MTM) with one working tape and an additional separate read-only input tape. In other literature [21], an MTM is often defined in such a way that the input is given on the first working tape.

It was obvious that GPT-5 had based its answer on such a model and was therefore unable to answer satisfactorily. As a consequence, it received 0.5 out of 2 points for question 4a, and 1 out of 3 points for question 4b. A second example is question 1a in the final:

Final: Question 1

Consider the language

$$L_{\geq 3} = \{\text{Code}(M) \mid M \text{ is a TM that accepts at least 3 words}\}.$$

(a) Show that $L_{\geq 3} \leq_{\text{m}} L_{\text{U}}$ by providing an explicit reduction and proving that your reduction is correct.

Here, Code(M) refers to the encoding of a Turing machine (TM) and $\leq_{\text{m}}$ refers to a *many-one reduction*. The answer provided by GPT-5 contained a valid many-one reduction, but the wrong language L_{U}. While students were familiar with L_{U} denoting the *universal language* $\{\text{Code}(M)\#w \mid w \in L(M)\}$, GPT-5 apparently interpreted it as $\{\text{Code}(M) \mid L(M) \neq \emptyset\}$.

Mathematical Notation. We copied the output of GPT-5 as it was provided and tried not to correct mathematical notation based on our assumptions. Often, the output of formulae was given in a manner that could not be interpreted by the TAs as valid. As an example, consider question 3 from the midterm:

Midterm: Question 3

Show that the language

$$L := \{0^i 1^j 0^k \mid i, j, k \in \mathbb{N}, i \neq j \text{ or } i \neq k \text{ or } j \neq k\}$$

over the alphabet $\{0, 1\}$ is non-regular.

The answer provided by GPT-5 started from a language denoted by $U = 010^*$ [sic]. This was not standard notation from the course. The proof that followed was essentially correct under the interpretation of $U = \{0^i 1^j 0^k \mid i, j, k \in \mathbb{N}\}$. As the model did not define U explicitly as part of its solution, this was not judged as clear by the grading TA.

Unanswered Questions. As mentioned above, parts of both exams consisted of multiple-choice questions.

For two of these questions, a significant amount of time after asking them, the

connection to the server was lost and no answer was provided. We tried again in a new instance with the same result and decided to interpret this as no answer. The first of these questions was question 2b:

Midterm: Question 2b

Which of the following statements are *always* true (meaning they hold for all corresponding languages L), only *sometimes* true (meaning they hold for some languages, but not for all), or *never* true (meaning there are no languages, for which they hold)?

6. Let $L_1, L_2 \subseteq \{a, b\}^*$ be non-regular, then

   $$L = \{vw \mid v \in L_1, w \in L_2\}$$

   is also non-regular.

This statement is sometimes true and sometimes false, though it is slightly harder to find a pair of languages L_1 and L_2 such that it is false. The second such question was part of a set of simple true/false questions:

Midterm: Question 6

True or false:

2. Let M be a 1-tape-TM with $m \geq 4$ states. Then there exists an equivalent 2-tape-TM that only has the states q_0, q_{accept}, and q_{reject}.

The statement was meant to be true, though it may also hinge on the exact definition of a 1-tape-TM. Thus, this might also be a case of the first type of error discussed, namely *background knowledge*.

Question 2a-4 in the Midterm. This single question has to stand in a category of its own. It was again a simple true/false question, part of a series of questions where students were asked to decide whether a given language was regular or not. Students received one point for a correct answer, no points for an incorrect answer, and 0.5 points for a missing answer.

The question was undoubtedly the hardest question of the exam. Of the 473 students who participated in the midterm, only 58 answered it correctly; and a further 30 did not answer.

Midterm: Question 2a-4

Is the language L_4 regular?

$$L_4 := \{a^n w \mid w \in \{a, b\}^*, n \in \mathbb{N} \setminus \{0\} \text{ and } |w|_b \geq n\}$$

The language looks like a classic example of a non-regular language. However, it is actually regular, since there is no requirement that w has to start with b. In fact, it holds that $L_4 = \{aw \mid w \in \{a, b\}^*, |w|_b > 0\}$, which can be more easily recognized as regular. This was the only multiple-choice question in both the midterm and the final that GPT-5 answered incorrectly.

4.2 Errors Made by Azure GPT-4o

The errors made by GPT-4o are harder to classify than those made by GPT-5. It produced quite a large volume of material, much of which was judged by the grading TAs as more or less meaningless, echoing the findings of Ding et al. [6] that GPT-4o produces unjustified / misleading claims.

It is perhaps worth noting, however, that both questions for which GPT-5 provided no answer at all were answered incorrectly by GPT-4o. While GPT-4o did answer several other multiple-choice questions incorrectly, this may indicate that these questions were in fact difficult.

5 Discussion

Recall the two research questions we stated in Section 1: (RQ1) how well do LLMs, without any fine-tuning or special training, perform in *Theory of Computing* exams; and (RQ2) whether there are specific types of tasks where LLMs outperform humans and vice versa.

Regarding **RQ1**, the answer emerging from our results is that their performance is relatively good. Since the question asked about the possible performance, the models should be judged by the more successful of the two. This model, GPT-5, achieved a very good grade, though not the highest possible. As mentioned above, it received 31 out of 35 points in the midterm, and 29.5 of 36 points in the final; and therefore 85% of all available points—corresponding to a grade of 5.5 out of 6.

To answer **RQ2**, we use the question categories defined in Section 3.1: MULTIPLE-CHOICE, RECALL, and TRANSFER. We also refer to the specific points for individual questions shown in Figure 1.

In the MULTIPLE-CHOICE category (questions 2 and 6 in the midterm, questions 2 and 3a in the final), GPT-5 scored above average in all questions, especially in the final. In the midterm, it lost points in three of these questions, as discussed in

Section 4.1. GPT-4o was less consistent, scoring above average in some questions and below average in others.

In the RECALL category, for most questions, GPT-5 answered perfectly and GPT-4o answered above-average. The exceptions were questions 4a and 4b in the final, for which both models received below-average scores. These two errors fall into the *background knowledge* category discussed in Sections 4.1 and 4.2.

In the TRANSFER category (questions 1, 3, and 4 in the midterm, questions 1, 5, and 6 in the final), GPT-5 scored perfectly in four of the questions (questions 1 and 4 in the midterm, questions 5 and 6 in the final) and clearly below average in the remaining two (question 3 in the midterm, question 1 in the final). Of these two questions, its performance in question 1 in the final again appears to be due to missing *background knowledge* from the course as discussed in Section 4.1. GPT-4o performed very poorly in this category, reaching a slightly above-average score in only one question and scoring clearly below average in all others.

In summary, it appears that both models scored poorly relative to human students in questions where specific knowledge from the course was required (as opposed to general knowledge of the field). This is to be expected. In addition, GPT-4o struggled greatly with TRANSFER tasks, constructions, and proofs. However, GPT-5 in particular performed better than human students in MULTIPLE-CHOICE questions, though it did not answer perfectly.

This leads to a sobering conclusion: if the goal is to design a question that GPT-5 cannot answer as well as a student, the solution may not lie in the TRANSFER category—as might have been suspected some years ago—but in the *background knowledge* error category: questions can be made resistant to LLMs by asking for and building on information that the model simply does not have access to. However, this could presumably be circumvented by a malicious student with little effort.

6 Limitations

Our work has certain limitations. Most notably, our dataset consists of only 12 exercises (including sub-exercises). We would also like to emphasize once more that our goal was to establish a lower bound on LLM performance. As described in Section 4, it is reasonable to assume that many of the "mistakes" made by GPT-5 were due to its use of notation different from that introduced in the course (accounted for under the *background knowledge* category in Sections 4.1 and 4.2). A model specifically prompted with, or even trained on, the course material would presumably perform better.

It should be noted of course that many of the problem statements from both exams are somewhat standard, and variations surely appear in some training sets.

For instance, there are a few typical non-regular languages to which the *pumping lemma* is commonly applied. The corresponding non-regularity proofs thus usually follow a very similar structure and do not contain fundamentally novel techniques.

Due to the large number of students taking the exams, individual questions were typically graded not by a single person but by a small group of TAs. To maximize consistency, they worked closely together and discussed any edge cases among themselves and we have no reason to suspect any systematic differences between them.

While transcribing the answers of GPT-5, an error occurred. In question 4 of the midterm (see Section 3.1), we erroneously swapped two entries in a table, which resulted in an initial score of 4 out of 5 points. The error was discovered later. Since it was clear that the correct entries would have received full marks, we adjusted the score accordingly. We are confident that this is the only such error that happened in the entire process.

7 Conclusion and Future Work

We analyzed the performance of GPT-based LLMs in a *Theory of Computing* course under realistic conditions. Our results give us good reason to believe that our findings represent a rather loose lower bound, and that GPT-5 in particular would be able to obtain an excellent grade if fine-tuned.

In addition, the results obtained by GPT-based models may not be representative of the broader landscape of LLMs. It might be worth comparing the performance of other models such as Gemini or DeepSeek. Given the diverse landscape of LLMs, their respective strengths may also lie in different question types.

Even so, based on the results from this work, a deeper comparison between multiple-choice and open questions would be of interest. Since GPT-5 did not answer all multiple-choice questions correctly, specifically exploring this type of question relative to student performance might be interesting, especially given that open questions typically demand considerably more effort from students.

Acknowledgment

The authors would like to thank the teaching staff of the 2025 course *Theoretical Computer Science* at ETH Zurich and Hans-Joachim Böckenhauer for carefully proofreading the first draft of the paper.

References

[1] Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. 2021. Program Synthesis with Large Language Models. https://arxiv.org/abs/2108.07732

[2] Mislav Balunović, Jasper Dekoninck, Ivo Petrov, Nikola Jovanović, and Martin Vechev. 2025. MathArena: Evaluating LLMs on Uncontaminated Math Competitions. In *Proceedings of the 39th Conference on Neural Information Processing Systems (NeurIPS 2025)*. 31 pages.

[3] Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, Wei Ye, Yue Zhang, Yi Chang, Philip S. Yu, Qiang Yang, and Xing Xie. 2024. A Survey on Evaluation of Large Language Models. *ACM Transactions on Intelligent Systems and Technology* 15, 3, Article 39 (March 2024), 45 pages. https://doi.org/10.1145/3641289

[4] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training Verifiers to Solve Math Word Problems. *CoRR* abs/2110.14168 (2021).

[5] Paul Denny, Viraj Kumar, and Nasser Giacaman. 2023. Conversing with Copilot: Exploring Prompt Engineering for Solving CS1 Problems Using Natural Language. In *Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 1 (SIGCSE 2023)*. ACM, 1136–1142. https://doi.org/10.1145/3545945.3569823

[6] Ming Ding, Federico Soldà, Weixuan Yuan, and Rasmus Kyng. 2025. Assessing GPT Performance in a Proof-Based University-Level Course Under Blind Grading. *Bulletin of EATCS* 146 (2025).

[7] Ryan E. Dougherty, Matei A. Golesteanu, and Garrett B. Vowinkel. 2025. Large Language Models with Reasoning on Theory Course Exams. In *Proceedings of the 30th ACM Conference on Innovation and Technology in Computer Science Education V. 2 (ITiCSE 2025)*. ACM, 785. https://doi.org/10.1145/3724389.3730783

[8] Ryan E. Dougherty, Tim Randolph, Tzu-Yi Chen, Jeff Erickson, Matthew Ferland, Dennis Komm, Jonathan Liu, Timothy Ng, Ana Smaranda Sandu, Michael Shindler, Edward Talmage, and Thomas Zeume. 2024. A Survey of Undergraduate Theory of Computation Curricula in the United States. In *Working Group Reports on 1st ACM Virtual Global Computing Education Conference (SIGCSE Virtual-WGR 2024)*. ACM, 1–14. https://doi.org/10.1145/3708550.3730559

[9] James Finnie-Ansley, Paul Denny, Brett A. Becker, Andrew Luxton-Reilly, and James Prather. 2022. The Robots Are Coming: Exploring the Implications of OpenAI Codex on Introductory Programming. In *Proceedings of the 24th Australasian Computing Education Conference (ACE 2022)*. ACM, 10–19. https://doi.org/10.1145/3511861.3511863

[10] Matei A. Golesteanu, Garrett B. Vowinkel, and Ryan E. Dougherty. 2024. Can ChatGPT Pass a Theory of Computing Course?. In *Proceedings of the 2024 ACM Virtual Global Computing Education Conference V. 1 (SIGCSE Virtual 2024)*. ACM. https://doi.org/10.1145/3649165.3690116

[11] Carolin Hahnel, Jamie Gorson Benario, Hieke Keuning, Natalie Kiesler, Tobias Kohn, Dennis Komm, Colleen Lewis, Dominic Lohr, Brent Reeves, Jaromír Šavelka, Jacqueline Staub, and Christina Weers. 2026. GenAI in Programming Education: Hypes, Hoaxes, and Hopes. In *Generative AI in Programming Education (Dagstuhl Seminar 25311)*, Michelle Craig, Paul Denny, Natalie Kiesler, and James Prather (Eds.). Vol. 15. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 253–279. https://doi.org/10.4230/DagRep.15.7.253

[12] Emma Harvey, Allison Koenecke, and Rene F. Kizilcec. 2025. Don't Forget the Teachers: Towards an Educator-Centered Understanding of Harms from Large Language Models in Education. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems (CHI 2025)*. ACM, Article 1064, 19 pages. https://doi.org/10.1145/3706598.3713210

[13] Orit Hazzan and Yael Erez. 2025. Rethinking Computer Science Education in the Age of GenAI. *ACM Transactions on Computing Education* 25, 3, Article 26 (June 2025), 9 pages. https://doi.org/10.1145/3732792

[14] Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, Jie Liu, Lei Qi, Zhiyuan Liu, and Maosong Sun. 2024. OlympiadBench: A Challenging Benchmark for Promoting AGI with Olympiad-Level Bilingual Multimodal Scientific Problems. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics, Volume 1: Long Papers (ACL 2024)*. ACL, 3828–3850. https://doi.org/10.18653/v1/2024.acl-long.211

[15] Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring Mathematical Problem Solving With the MATH Dataset. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks*, Joaquin Vanschoren and Sai-Kit Yeung (Eds.). https://datasets-benchmarks-proceedings.neurips.cc/paper/2021/hash/be83ab3ecd0db773eb2dc1b0a17836a1-Abstract-round2.html

[16] Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. 2025. LiveCodeBench: Holistic and Contamination Free Evaluation of Large Language Models for Code. In *The 13th International Conference on Learning Representations*. https://openreview.net/forum?id=chfJJYC3iL

[17] Nero Li, Shahar Broner, Yubin Kim, Katrina Mizuo, Elijah Sauder, Claire A. To, Albert Wang, Ofek Gila, and Michael Shindler. 2025. Investigating the Capabilities of Generative AI in Solving Data Structures, Algorithms, and Computability Problems. In *Proceedings of the 56th ACM Technical Symposium on*

Computer Science Education V. 1 (SIGCSE 2025). ACM, 659–665. https://doi.org/10.1145/3641554.3701946

[18] Andre Fabiano Pereira and Rafael Ferreira Mello. 2025. A Systematic Literature Review on Large Language Models Applications in Computer Programming Teaching Evaluation Process. *IEEE Access* 13 (2025), 113449–113460. https://api.semanticscholar.org/CorpusID:279775486

[19] Mike Richards, Kevin Waugh, Mark Slaymaker, Marian Petre, John Woodthorpe, and Daniel Gooch. 2024. Bob or Bot: Exploring ChatGPT's Answers to University Computer Science Assessment. *ACM Transactions on Computing Education* 24, 1, Article 5 (Jan. 2024), 32 pages. https://doi.org/10.1145/3633287

[20] Yuhong Shi, Kun Yu, Yifei Dong, and Fang Chen. 2026. Large language models in education: a systematic review of empirical applications, benefits, and challenges. *Computers and Education: Artificial Intelligence* 10 (2026), 100529. https://doi.org/10.1016/j.caeai.2025.100529

[21] Michael Sipser. 2013. *Introduction to the Theory of Computation* (third ed.). Cengage Learning.

[22] Haoxiang Sun, Yingqian Min, Zhipeng Chen, Wayne Xin Zhao, Lei Fang, Zheng Liu, Zhongyuan Wang, and Ji-Rong Wen. 2025. Challenging the Boundaries of Reasoning: An Olympiad-Level Math Benchmark for Large Language Models.

[23] Alan M. Turing. 1950. Computing Machinery And Intelligence. *Mind* LIX, 236 (10 1950), 433–460. https://doi.org/10.1093/mind/LIX.236.433

[24] Zihan Wang, Jiaze Chen, Zhicheng Liu, Markus Mak, Yidi Du, Geonsik Moon, Luoqi Xu, Aaron Tua, Kunshuo Peng, Jiayi Lu, Mingfei Xia, Boqian Zou, Chenyang Ran, Guang Tian, Shoutai Zhu, Yeheng Duan, Zhenghui Kang, Zhenxing Lin, Shangshu Li, Qiang Luo, Qingshen Long, Zhiyong Chen, Yihan Xiao, Yurong Wu, Daoguang Zan, Yuyi Fu, Mingxuan Wang, and Ming Ding. 2025. AetherCode: Evaluating LLMs' Ability to Win In Premier Programming Competitions. https://arxiv.org/abs/2508.16402

[25] Sarah J. Zhang, Samuel Florin, Ariel N. Lee, Eamon Niknafs, Andrei Marginean, Annie Wang, Keith Tyser, Zad Chin, Yann Hicke, Nikhil Singh, Madeleine Udell, Yoon Kim, Tonio Buonassisi, Armando Solar-Lezama, and Iddo Drori. 2023. Exploring the MIT Mathematics and EECS Curriculum Using Large Language Models. https://arxiv.org/abs/2306.08997