Using Benchmarking Infrastructure to Evaluate LLM Performance on CS Concept Inventories: Challenges, Opportunities, and Critiques

MURTAZA ALI, University of Washington, USA
PRERNA RAO, University of Washington, USA
YIFAN MAI, Stanford University, USA
BENJAMIN XIE, Stanford University, USA

BACKGROUND AND CONTEXT. The pace of advancement of large language models (LLMs) motivates the use of existing infrastructure to automate the evaluation of LLM performance on computing education tasks. Concept inventories are well suited for evaluation because of their careful design and prior validity evidence.

OBJECTIVES. Our research explores the feasibility of using an automated benchmarking framework to evaluate computer science (CS) concept inventories. We explore three primary objectives: evaluation of LLM performance on the SCS1 and BDSI concept inventories; informal expert panel review of items which had variations between LLM and expected student performance; and description of challenges with using benchmarking infrastructure as a methodological innovation.

METHOD. We used the Holistic Evaluation of Language Models (HELM) framework to evaluate the SCS1 and BDSI against 10 LLMS with zero-shot and few-shot in-context learning: GPT (3.5, 4.0), Claude (1.3, 2.0, 2.1), Llama (7B, 13B, 70B), Mistral v0.1 7B, and Mixtral 8x7B. We used psychometric data from prior studies to measure knowledge levels for each LLM run. We then conducted an informal expert review to qualitatively explore how question design, CS content knowledge, and LLM design may explain differences between LLM and expected student performances.

FINDINGS. Our quantitative analysis found that most LLM response patterns reflected a below average introductory computing student with the SCS1 and did not fit the psychometric 2PL model for the BDSI. Our qualitative analysis identified that LLMs performed well on code infill questions, but poorly on nested conditionals, runtime analysis, and longer questions. We also identified several methodological challenges related to item security, translation, the structure when using HELM.

IMPLICATIONS. We consider the feasibility of using automated benchmarking as a methodology to support more reproducible, replicable, and rigorous investigations to understand the intersection of LLM capabilities, computing concepts, and assessment design. We also consider connections between psychometric approaches and LLM evaluations to inform the design of computing assessments that are more resilient to LLM advancements.

CCS Concepts: • **Social and professional topics** → **Computing education**; • **Human-centered computing** → *Human computer interaction (HCI)*.

Additional Key Words and Phrases: computing education, large language models, benchmarking, psychometrics, concept inventories

ACM Reference Format:

Murtaza Ali, Prerna Rao, Yifan Mai, and Benjamin Xie. 2024. Using Benchmarking Infrastructure to Evaluate LLM Performance on CS Concept Inventories: Challenges, Opportunities, and Critiques. In *ACM Conference on International Computing Education Research V.1*

Authors' addresses: Murtaza Ali, mali53@uw.edu, University of Washington, Human-Centered Design and Engineering, Seattle, WA, USA; Prerna Rao, prernar@uw.edu, University of Washington, Human-Centered Design and Engineering, Seattle, WA, USA; Yifan Mai, maiyifan@stanford.edu, Stanford University, Center for Research on Foundation Models, Stanford, CA, USA; Benjamin Xie, benjixie@stanford.edu, Stanford University, Institute for Human-Centered Artificial Intelligence, McCoy Family Center for Ethics in Society, Stanford, CA, USA.

Manuscript submitted to ACM

(ICER '24 Vol. 1), August 13–15, 2024, Melbourne, VIC, Australia. ACM, New York, NY, USA, 26 pages. https://doi.org/10.1145/3632620.3671097

1 INTRODUCTION: THE RAPIDLY EVOLVING LANDSCAPES OF LARGE LANGUAGE MODELS

Large Language Models (LLMs) are advancing rapidly and have caught significant public attention [13, 71]. These models have hundreds of billions of parameters and train on broad datasets to enable adaptability to a wide range of downstream tasks, such as text and image generation [13]. These models also have several educational applications, ranging from assisting with grading [85] to generating open-ended practice exercises [110]. Furthermore, since LLM training data includes extensive code, researchers have also explored the utility of LLMs specifically in computing education [87].

Timely evaluations of new LLM capabilities on computing education tasks, such as performance on assessments [98] and ability to generate code explanations [66, 93]), could inform the practice of computing education instructors and researchers alike. However, the pace of advancement in LLM performance make them difficult to rigorously evaluate through typical scholarly means. In the 100 days since OpenAI released its latest generation LLM, GPT-4 Turbo[1], 29 different organizations released 35 new LLMs [99]. By contrast, computing education research publications typically publish annually for conferences (e.g. ICER, SIGCSE, ITiCSE, Koli Calling) and quarterly for journals (e.g. TOCE, CSE). Evaluations of LLM performance risk outpacing evaluations of their capabilities and affordances within computing education contexts.

In response to the velocity of advancement and the broad adaptability of LLMs, Artificial Intelligence (AI) researchers have resorted to developing *benchmarks* to evaluate LLM performance. These benchmarks are standardized tasks that attempt to measure LLM capabilities across a broad spectrum, including consideration of social biases [23, 79], mathematical problem solving, [48], and medical exams [40]. Joint efforts to develop and evaluate benchmarks have resulted in collaborative benchmarks such as the Beyond the Imitation Game (BIG-bench), which includes 204 tasks developed by 450 authors from 132 institutions [9]. Task topics are diverse, drawing upon problems from linguistics, social bias, math, software development, and beyond.

Automating certain evaluations of LLM capabilities could enable more capacity to conduct more open-ended and human-centered evaluations. Echoing Baker [5], the goal of LLMs in computing education is not to use the most effective LLMs, but to find effective ways to develop knowledgeable and successful (human) learners. Within computing education research, this includes broadening participation in computing, preparing educators, developing quality curricular materials and assessments, and scaling efforts [10]. We argue that we cannot, and should not, automate evaluations of the impact of LLMs on these efforts. What we *can* do is automate some *closed-ended* evaluations (e.g. LLM performance on an assessment) to encourage deeper, more open-ended analyses (e.g. impacts of LLM usage on broadening participation) and thus support more *rigorous* overall evaluations of LLM performance. Within the context of computing education, closed-ended evaluations could include LLM performance of *concept-inventories*, assessments designed to determine whether a student has accurate working knowledge of a specific set of computing concepts (e.g. introductory computing knowledge [82] or data structures [86]).

This paper explores the feasibility of using an automated benchmarking framework, Holistic Evaluation of Foundation Models (HELM) [61] to support more reproducible, replicable, and rigorous evaluations of LLM capabilities in computing education. *Reproducibility* refers to the ability to achieve the same findings as a prior study using existing data that

[1]released 5 November 2023 [99]

prior study [103], and *replication* refers to following the design of a prior study with newly collected and analyzed data to determine if a new study yields the same findings as a prior study [103].

Prior work in CER include literature reviews of empirical studies, replication studies, and infrastructure to improve reproducibility. In a systematic literature review of computing education research literature, Heckman et al. [38] found that over 80% of reviewed papers included some form of empirical evaluation, with quantitative evaluation methods being most frequently reported. Examples of prior replication studies in CER include investigating subgoal labeling [77] and designing a new language-agnostic knowledge assessment [82] in CS1. Replication studies that fail to replicate previous findings are also useful (e.g. [58, 72]), as they can identify alternative explanations for prior findings. To support more reproducibility and replication in CER, prior work has explored pre-registering studies Brown et al. [14]. More reproducible and replicable empirical studies can result in knowledge building through meta-analyses [74] and theory building [70, 80].

We focus on the performance of 10 LLMs on two introductory CS concept inventories: the SCS1, which measures introductory computing (CS1) knowledge [82], and the BDSI [86], which measures knowledge of data structures. We chose to evaluate concept inventories because they have strong validity evidence and likely were not used to previously train LLMs, overcoming two common issues with existing benchmarks [60, 91, 115]. We compared LLM performance with expected student performance by using psychometric properties reported from prior studies. This comparison enabled us to identify unusual items, which we define as items for which LLM and expected student performance deviated (e.g., LLMs performing unusually well on difficult questions). We then conducted a qualitative review with computing education, LLM, and assessment design experts to understand how LLM design, item design, and computing concepts may affect LLM performance when compared to expected student performance. By doing so, we explored the following research questions:

(1) **How do LLMs perform on the SCS1 and BDSI concept inventories?**
(2) **How do AI, computing education, and assessment experts interpret deviations in LLM capabilities from expected student performance?**
(3) **What are challenges with using benchmarking infrastructure to automatically evaluate LLMs?**

In this paper, we contribute a feasibility study into the use of automated benchmarking infrastructure as a methodological innovation for computing education. This feasibility study includes 1) a mixed-methods empirical evaluation to evaluate the performance of 10 LLMs on two CS concept inventories in comparison to expected student performance and 2) a description of methodological challenges in leveraging benchmarking infrastructure to automate LLM evaluations for computing education research.

2 RELATED WORK

In this section, we describe related work pertaining to the opportunities and challenges with evaluating LLMs with benchmarks. We then discuss computing education research on concept inventories (CIs) and their validity evidence. Collectively, this prior work situates our study in using HELM [61] as automated benchmarking infrastructure to evaluate LLM performance on two concept inventories (SCS1 [82] and BDSI [86]).

2.1 Benchmarking Large Language Models (in Computing Education Research)

A language model (LM) aims to model the generative likelihood of word sequences—to predict the probability that one word follows another in a stream of text [114]. LM research has advanced significantly over the years, with the

following progression: 1) statistical models (SLMs) [46], 2) neural language models (NLMs) [8], 3) pre-trained language models (PLMs) [59], and, most recently, 4) large language models (LLMs) [114]. These models, broadly speaking, are often considered to fall under the broad class of *foundation models*, an umbrella term for any model trained on broad data and capable of being fine tuned for specific tasks [13].

From a technical standpoint, LLMs are simply very large PLMs; this upsizing is achieved by scaling the model or the data size and results in better performance on several tasks. One such improvement is the chief source of the current fascination with LLMs across domains: LLMs enable applications which possess a remarkable ability to carry on realistic dialogue with humans [114]. The most popular example of such an application is ChatGPT [109].

LLMs are often evaluated via comparison with existing benchmarks. However, there is still the question of *how* LLMs are actually evaluated (i.e., what measures/metrics are used to evaluate them). In their survey on evaluating LLMs, Chang et al. [16] note that evaluation can be split into two categories: automatic evaluation and human evaluation. Automatic evaluation is more efficient and standardized, generally focused on measuring LLM performance via the following four metrics:

(1) **Accuracy**: How well a model performs a predefined task, as measured by autmated metrics such as F1 score or Exact Match.
(2) **Calibration**: The level of agreement between the model's confidence level and its actual accuracy.
(3) **Fairness**: A measure of whether the model is consistent across various group attributes.
(4) **Robustness**: How well a model performs against challenging inputs and adversarial attacks. In some contexts, including within HELM, robustness can also include a model's ability to perform well even with input transformations and/or perturbations.

Human evaluation is more time consuming and subjective, but can lead to more reliable evaluations for tasks that are hard to standardize, such as those that involve open generation [84]. Chang et al. [16] outline the following six human assessment criteria for LLM evaluation:

(1) **Accuracy**: In this case, accuracy refers to the precision and correctness of the output as judged by human experts, as opposed to calculation via automated metrics.
(2) **Relevance**: The appropriateness and significance of the output.
(3) **Fluency**: How smoothly the model's output reads, including syntax, semantics, and tone/style.
(4) **Transparency**: How well the model explains its reasoning.
(5) **Safety**: The model's ability to refrain from producing harmful or inappropriate content.
(6) **Human alignment**: How well the model produces output that is in line with a human's expectations and values. This criterion seeks to ensure, at a high level, positive interactions with human users.

An alternative dimension on which LLM evaluation can be considered is open-ended vs. closed-ended evaluations. Open-ended evaluations consider an LLM's proficiency at tasks that do not have a single correct answer, such as writing open-ended story content or engaging in critical reasoning. Closed-ended evaluations involve tasks that can be graded without judgment. A helpful analogy here is to consider the difference between grading a multiple-choice exam (closed ended) and an argumentative essay (open ended); both serve their purposes, but are fundamentally different.

Within CS Education LLM research, different types of evaluations have been used for different research goals. Researchers have engaged in open-ended analyses to evaluate automatically generated programming exercises [22], context-aware error explanations [92, 100], code explanations [94], and student code feedback [6]. For tasks that simply involve evaluating GPT's performance on an assessment [97] (especially a multiple-choice one), researchers often use a

closed-ended evaluation followed by a deeper, open-ended analysis [30, 96]. We advocate for this dual approach in our work. Stopping at a closed-ended, automatic evaluation for assessment instruments does not provide insight into *why* an LLM performs the way that it does, and it misses the opportunity to learn about *how* educators might design assessment materials that are more resilient to accurate completion by LLMs. We argue in favor of moving from only a closed-ended, automatic evaluation of assessment instruments to one that also includes open-ended, human evaluation, and in our paper we demonstrate one method of doing so.

2.1.1 Critiques of Benchmarking. We frame critiques of current automated benchmarking practices from the perspectives of machine learning, psychometrics, and value tensions.

A machine learning concern with benchmarks relates to *benchmark leakage*. This term generally refer to when a LLM trains on data that we want to evaluate it against, such as training on benchmarks. This violates a fundamental idea of machine learning, that a model should never train on the test set. Benchmark leakage refers to data or tasks used for evaluation or test sets that are used for model training [60, 91, 115]. Benchmark leakage includes *test data contamination* in various forms: the inclusion of test data guidelines (e.g. information about a concept inventory), examples (e.g. concept inventory questions without solutions), and/or labels/annotations (e.g. concept inventory questions annotated with solutions) in the pre-training data [60, 115]. Benchmark leakage may also include *task contamination*, the inclusion of task training examples in pre-training data, such as examples of concept inventory question structures [60]. This may invalidate zero or few-shot in-context learning approaches, but may also be acceptable when the structure of a task is not part of the evaluation (e.g. wanting an LLM to train on the structure of multiple choice questions to evaluate computing knowledge using multiple choice questions) [24]. Recommendations to address benchmark leakage include benchmarks that reflect broader capabilities, data decontamination checking by LLM developers, disclosure of training datasets, and more diverse sets of test prompts for benchmark maintainers [60, 91, 115]. However, the inclusion of benchmarks in training data for LLMs is an ongoing concern with their continued use.

A psychometrics-related concern with benchmarks involves validity. There are multiple perspectives of validity according to educational statistics and psychometric literature [51, 75, 76]. A common framing of validity defines it as "an overall evaluative judgment of the degree to which empirical evidence and theoretical rationales support the adequacy and appropriateness of interpretations and actions on the basis of test scores or other modes of assessment" [75, 76]. Related to benchmarking, validity is an evaluative summary of evidence for how people can interpret and use benchmark results [51, 76]. Raji et al. [89] has argued that benchmarks lack construct validity [75, 76], evidence that LLM performance on benchmarks reflects processes, strategies, and knowledge of a latent construct (e.g. computing knowledge). A particular concern relates to *construct-irrelevant variance* [76], where a benchmark is too broad and as a result, score variance is associated with confounding constructs or other properties irrelevant to the intended construct. The broad applicability of LLMs makes it difficult to consider the validity of interpreting and using benchmark results, in part because latent constructs of interest (e.g. "common tasks" [89], bias, programming knowledge) are underspecified and often context dependent. For example, benchmarks to evaluate gender bias in LLMs (e.g. [35, 52, 83]) tend to focus on binary genders in relation to occupational stereotypes (e.g. how stereotypes often associate doctors as being men and nurses as being women). This consideration of only two genders within occupational contexts is largely because of data availability [16, 52]. However, gender is a fluid social construct that can be context dependent (e.g. a nonbinary person coming out to close friends, but not to workplace colleagues or school classmates) [31]. Therefore, construct-irrelevant variance may occur when considering gender bias in LLM outputs beyond occupational contexts (e.g. in educational or social contexts) and when considering more than two genders.

An ethical concern with benchmarks relates to the values embedded into designing benchmarks. Dehghani et al. [21] identified four biases that impact LLM researchers benchmarks: task selection bias (dependence of model performance on tasks and datasets selected for benchmarks), community bias (effects of community pressures to develop and emphasize certain benchmarks), statefulness of benchmarking (decisions made in developing new models are informed by the errors and successes of previous models on the same benchmarks, and rigging (selecting evaluation methods and metrics that best fit given model and resource constraints). Blili-Hamelin and Hancox-Li [11] drew connections between ethical tensions of LLM benchmarking and IQ tests [28]. These ethical concerns related to task-selection, narrowly defined standards for construct validity, and positive feedback loops between benchmarks and types model development.

2.1.2 LLMs in Computing Education Research. LLMs have swiftly been incorporated into computing education research within the last year, with researchers exploring their potential for various tasks including enhancing error messages [56], generating code [81], generating code explanations [57, 65], and simulating students using LLMs [1]. Some researchers have even explored the potential of LLMs to directly generate advanced educational materials, such as with Agrawal et al. [2]'s CyberGen system.

In their 2023 report on the expanding use of generative AI in computing education research, Prather et al. [87] identified 71 articles related to the topic, with over three-fourths published in 2023 itself. Papers centered around 5 primary topics: 1) assessing the performance of LLMs, 2) position papers/surveys/interviews, 3) interactions between programmers and LLMs, 4) use of LLMs to review student work, and 5) use of LLMs to generate educational resources and materials. In their report, the authors additionally build upon existing research by benchmarking several generative AI models on a group of computing education data sets.

Specifically with respect to benchmarking in computing education *within generative AI*, researchers published a wealth of impactful findings in 2023 and 2024, attempting to keep up with the advancements of language models and their effect on performance in various areas. Mahon et al. [68] investigated GPT-4's performance on standardized national exams for high school computer science in Ireland and the UK, finding that GPT-4 performed quite well overall, but struggled with questions involving images or symbols. Savelka et al. [96] found a strong rate of improvement when benchmarking GPT-4 (as compared with previous generations of the model) on three Python courses containing everything from simple multiple-choice questions to involved free-coding questions spanning multiple files. Prather et al. [87] also report on benchmarking as part of their above paper, focusing on the task of generating a solution to a programming problem and reviewing a wide range of available data sets to categorize the content they contained [87]. Finally, they replicate one of the first studies on LLMs in computing education [30] using GPT-4, GPT-3.5 Turbo, and CoPilot on the original data, as well as on two previously untested data sets, the Automated Programming Progress Standard (APPS) [39] and FalconCode [20]. Like [96], they found significant improvements with newer models while also reporting on challenges with LLMs parsing question formats and adding additional output which was not specified in the problem description.

Prather et al. [87] also noted two challenges to benchmarking that are particularly relevant to our work: 1) It can be difficult to meaningfully benchmark LLMs because of the speed at which new, more capable models are released, and 2) because papers often use a wide variety of prompts and evaluation approaches without delving into the details, it can be difficult to validate results. Our methodology addresses the first concern via our use of HELM: Because HELM is consistently updated with the latest LLMs, integrating CIs into the framework will allow new model benchmarks to be computed more efficiently. To provide the opportunity for replication as well as encourage further work, we describe our evaluation methodology in great detail in Section 3.

2.2 Concept Inventories

A *concept inventory* (CI) is a standardized assessment used to measure student understanding of core concepts in a discipline, designed specifically to elicit information about *misconceptions* [101]. CIs serve to identify areas of conceptual difficulty prior to instruction and evaluate the impact of pedagogical interventions in students' conceptual understanding [63]. Concept inventories are typically involve multiple choice items/questions with one correct solution. All other solutions are incorrect and collectively referred to as *distractors*. A well-designed CI item has distractors which each reflect a common misconception about a concept; thus, based on the answers students select, instructors can identify student misconceptions about the material [63].

Concept inventories date back to 1992, with the advent of the Force Concept Inventory (FCI, [44]). At the time, physics students held several misconceptions about Newtonian force which physics courses were not correcting [36]. In the years following its release, the FCI led to revolutionary changes in physics education (e.g. peer instruction [18]) and thus precipitated the adoption of CIs across STEM fields [7, 26, 64, 78].CI work reached computer education nearly two decades later, starting with introductory work identifying misconceptions [33, 50]. This preliminary work led to the publication of the first two empirically validated CIs in computer science: the Digital Logic Concept Inventory (DLCI) and the Foundational Computer Science 1 Assessment (FCS1) [42, 102].

With this foundation laid, CI work in computer science slowly began to expand. The first literature review studying such work was conducted in 2014 by Taylor et al. [101], who documented several important points detailing the development of research work:

- Though more researchers were beginning to explore the space, progress was still limited in 2014. Taylor et al. recorded 6 total CIs in computer science, and among these only the FCS1 and DLCI were fully developed and validated.
- Misconceptions were common among introductory computer science students, and so CI research is important and worth expanding.
- Taylor et al. noted several challenges in building CIs unique to computer science, revolving around the fact that computer science is a young and fluid field (especially in comparison to a natural science such as Newtonian physics) with variations in how it is taught. For example, many courses teach the same concepts in different programming languages. Do these warrant distinct CIs? Are important, more "overarching" skills in computer science–such as debugging and program design–appropriate for inclusion in a CI? Their documentation of these challenges went on to explicitly and implicitly influence future work in this space.

In the years following Taylor et al.'s review, the speed of computer science CI research increased substantially. In 2023, Ali et al. published a systematic review with the following updated findings [3]:

- As of 2023, there were 33 total computer science CIs in existence, 12 of them validated.
- Taylor et al.'s prediction of potential challenges proved prescient. Though researchers rarely set out with the intention of addressing these challenges, they nearly always presented themselves, and as such, progress toward potential solutions was made. For instance, one novel solution to varied programming languages was to develop a method for transitioning a CI from one language to another [15].
- Research also expanded beyond simply building new CIs using established methods, with some researchers actively exploring novel methodologies for CI development itself [90, 107].

CI work in computing education continues to expand today. By incorporating CIs as benchmarks/scenarios in HELM, we hope to make evaluation of future LLM capabilities more consistent with evaluations on other LLMs and timely.

2.2.1 Validity Evidence and Psychometrics. Item Response Theory (IRT) methodologies are foundational to psychometrics [19]. IRT provides more sample-agnostic statistics which estimate learner knowledge and question properties separately using falsifiable models. IRT enables the estimation of question-level and test-level parameters, as well as test-takers' knowledge levels. This provides estimates of the difficulty and discrimination of each question for learners of different knowledge levels. It does this by estimating the correspondence between unobserved latent variables (e.g. learners' computing knowledge, difficulty of questions) and observable evidence of knowledge (e.g., people's responses to questions). By fitting response data to a model (e.g. logistic model), we can estimate question parameters (e.g. difficulty) with fewer assumptions about the characteristics of the sample. We can also make predictive statements about learner performance based on knowledge level. By doing so, estimates of test-taker, item, and assessment properties generalize beyond the specific sample of test-takers.

A key principle of IRT is placing test-takers and questions on the same normally distributed, typically unidimensional continuum. The center of 0 represents the knowledge level for the average test-taker of the population. An average test-taker would have a 50% chance of getting an average question correct. Therefore, if a test-taker's predicted knowledge level is greater than the difficulty of the question, then they are more likely to get the question correct. This continuum helps model the relationship between a learner's latent knowledge level and their observed item performance as a monotonically increasing function.

In this paper, we focus in particular on the 2 Parameter Logistic (2PL) model because prior work fit responses to the SCS1 [112] and BDSI [86] to the 2PL model. For the 2PL model, each item has two parameters: *difficulty* and *discrimination*. The estimated difficulty of a question is the knowledge level at which a test-taker has an equal chance (50%) of answering a question correctly and incorrectly. Discrimination is how well a question distinguishes between test-takers of varying knowledge levels. A high discrimination value is desireable because it indicates that the probability of a test-taker answering correctly changes significantly based on their knowledge level. With discrimination, we can see if questions at the same difficulty level provide more or less information about test-takers' knowledge.

3 METHOD

To answer our research questions, we automatically evaluated the concept inventories with HELM. We then conducted expert review [73] on items with LLM performance deviating from expected student performance. We describe our methodology for selecting CIs and evaluating CIs with HELM. We then describe our mixed-methods analysis involving modeling LLM performance with IRT and reviewing unusual items with AI, computing education, and psychometric experts.

3.1 Concept Inventory Selection & Item Translation

Authors of CIs generally do not make them publicly available, as widespread access to the test items would interfere with the instrument's validity. If a student saw the questions beforehand, they could simply memorize the answers, and the CI would lose its ability to ascertain the student's misconceptions. In order to obtain the CIs of interest, we reached out to each CI's author team independently.

The full list of CIs we attempted to obtain was drawn from the validated list of CIs from Ali et al. [3], along with two additional CIs on Cybersecurity [41] and the Rust Programming Language [17] which were validated after the publication of Ali et al. [3]. We successfully obtained seven CIs, including two and omitting five.

The two CIs we included in this study were the Second Computer Science 1 Assessment (SCS1, [82]) and the Basic Data Structure Inventory (BDSI, [86]). The SCS1, properly known as the Second CS1 Assessment, is a CI developed in

2016 which focuses introductory computer science concepts [82]. Formally, the SCS1 is a replicated CI, isomorphic to the earlier FCS1, or Foundational CS1 Assessment [102]. Parker et al. [82] developed the SCS1 to enable free use by a broad research community, arguing that replication of assessments is important to addressing inbuilt issues and adding extensions. The SCS1 is an assessment instrument which has validity evidence from multiple independent studies [112]. Conceptually, it is suited for computer science students who have completed an introductory programming course. Additionally, it makes use of a pseudocode language to enable accessibility to students with backgrounds in different programming languages. The BDSI, or Basic Data Structures Inventory, was released in 2019 and makes use of an extended version of the SCS1 pseudocode language to assess students' proficiency in data structures [86]. It tests concepts at a level above the SCS1, and it is suited for students who have completed a CS2 course in computer science, generally connoting a course that introduces students to data structures [43].

We excluded the five other CIs for the reasons stated below:

- Basic Recursion CI [37]: Unlike most CIs, the Basic Recursion CI consists of completely open-ended questions. This presented difficulties in automatically checking for correctness within HELM.
- BST and Hash Tables CI [53]: This CI consists of several images and graph structures; determining how to best represent these to LLMs that do not support images directly will require further work. Our work provides preliminary insight into this, as the BDSI also included graphs, but to a much lesser extent.
- CCI (Cybersecurity CI) [41]: The CI authors did not consent to inclusion of instrument because of similar ongoing research.
- First-Year Computer Science CI [104]: This CI is currently only available in German, which is not as comprehensible as English to the majority of LLMs; this would have confounded the results.
- MG-CSCI (Middle-Grades Computer Science CI) [88]: This CI consists primarily of questions in the blocks-based programming language Snap!, represented as images. Thus, we faced a similar issue as with the BST and Hash Tables CI in representing the questions.

3.2 Integrating CIs into HELM

To systematically evaluate LLM capabilities to answer the SCS1 and BDSI concept inventories, we used the automated benchmarking infrastructure provided by the Holistic Evaluation of Language Models (HELM). HELM is an open-source LLM benchmarking framework built by Stanford's Center for Research on Foundation Models (CRFM). HELM provides benchmarking infrastructure for nearly 150 LLMs and includes over 100 benchmarks, and provides researchers with the ability to add new models and benchmarks as work progresses [61]. [2].

To add a benchmark to HELM, one must implement a HELM Scenario subclass in Python, which defines how data is imported, parsed, and converted into HELM Instance objects. A single Instance object represents a CI question and its associated answer choices. For our work, we implemented a new Scenario within HELM, CIMCQAScenario, which handles the conversion of a JSON file representing a CI into a list of Instances that can be passed into HELM's core program. This infrastructure enabled consistent prompt formatting for single-option multiple-choice questions across multiple benchmarks.

Because the CIs we worked with were initially only available as PDF or Microsoft Word files of questions and answers, we structured them into JSON files as a preprocessing step. In doing so, we made a number of decisions concerning

[2]Number of LLMs, scenarios, metrics as of March 2024. https://crfm.stanford.edu/helm/classic/latest/

how to represent code indentations, graph structures, tables, and other aspects of the CIs that did not have immediately clear text representations. We made the following key decisions:

- Line breaks in code are indicated with a newline character.
- One level of indentation is represented via four space characters.
- Questions/answers in table format are represented by listing each row individually, with the column title preceding each individual entry.
- Any tree diagrams in the questions are represented with textual descriptions.
- Questions with multiple correct answers (specifically, two questions on the BDSI) are excluded, as HELM does not currently support this answer format.

A more detailed document outlining the design decisions made for each CI is available as supplemental material to this paper.

We attempted to prevent LLM creators from using CIs to train future models but largely failed to do so. The only LLM developer that provided the capabilities to restrict data usage at the time of this study was OpenAI. However, this required an Enterprise account, which we did not have. We did avoid publicly sharing the CI questions and answers as a published benchmark. This mitigated the risk of test data leakage (CI questions and solutions cannot be scraped from the web), but did go against typical practices of publishing benchmarks for others to interrogate and use [9]. We obtained verification from SCS1 and BDSI maintainers that we had permission to use their CIs prior to running any CI questions through an LLM.

Finally, in order to enable few-shot runs of the model, we developed a set of in-context learning examples [24]: 5 for the SCS1 and 3 for the BDSI. These examples are not used to fine tune models for better performance; rather, they assist HELM in familiarizing itself with the correct input-output structure for evaluation items. This helps reduce situations where an LLM response is incorrectly evaluated due to a minor variation in the desired output format.

We chose to evaluate the following 10 LLMs created by four organizations:

(1) Anthropic Claude v1.3, Anthropic Claude 2.0, and Anthropic Claude 2.1 (by Anthropic)
(2) GPT-3.5 Turbo (0613) and GPT-4 (0613) (by OpenAI)
(3) Llama 2 (7B), Llama 2 (13B), and Llama 2 (70B) (by Meta)
(4) Mistral v0.1 (7B) and Mixtral (8x7B 32K seqlen) (by Mistral AI)

These models reflect a subset of models from HELM Lite [62] that were available at the time.

3.3 Analysis

3.3.1 Quantitative Analysis: Psychometric Properties to Compare LLM and Student Performance. We used difficulty and discrimination parameters defined in prior work ([86, 112]) to fit the SCS1 and BDSI to separate 2PL models. We then checked whether LLM response patterns were consistent with each 2PL model. We checked the person-fit statistic l_z, a standardization of the test-taker log likelihood function L to address the interaction of $ln(L)$ and θ [19, 25, 45]. l_Z is standardized, so a value of 0 denotes a perfectly expected or typical response pattern. Values above 2.0 could indicate overfitting (unexpectedly good fit) and below -2.0 could indicate noisy or unexpectedly poor fitting [45]. If the person-fit statistic was acceptable ($|l_z| < 2.0$), then we reported the latent knowledge level θ of each LLM run, effectively treating each one as independent test-takers and answering our first research question. θ is normalized, with 0 denoting an "average" test-taker, > 0 denoting an above average test-taker, and < 0 denoting a below average test-taker [19].

3.3.2 Qualitative Analysis: Informal Expert Panel Review. We identified unusual items in which LLM performance on these items deviated from expected student performance. To calculate expected student performance, we used the difficulty parameter (α) from the 2PL model for each item. α is effectively a z-score [19], with the proportion of the normal curve above that value representing the proportion of students with a $\geq$ 50% expected probability of getting the item correct. We then compared that percentage to the proportion of LLMs that got an item correct.

We considered two kinds of unusual items: those with high difficulty and those with low difficulty. Unusual high difficulty items are those with the greatest difficulty that had a greater proportion of LLM runs getting them correct when compared to the expected proportion of students. Therefore, unusual high difficulty items are those in which LLMs performed unexpectedly well. Unusual low difficulty items are those with the lowest difficulty in which the proportion of LLM runs getting them correct is less than the expected proportion of students getting it correct. Therefore, unusual low difficulty items are those in which LLMs performed unexpectedly poorly. We considered at most three high difficulty and low difficulty items that fulfilled this criteria.

Three authors participated in reviewing unusual items. These authors had experience in computing education research (three authors had collectively published over 12 papers to computing education research venues), teaching higher education computing courses (two authors had served as instructors; all four had experience as teaching assistants), psychometrics (one author had previously published multiple papers related to educational statistics and assessment design, and one author had developed materials for machine learning courses taught internationally), and LLMs (one author had published multiple papers on benchmarking and had industry experience developing deep learning algorithms).

The goal of this informal expert panel review [54] was to understand how LLM design, assessment design, and/or computing knowledge may explain deviations in LLM and expected test-taker performances. For each unusual question, experts considered whether the original question design, computing concept it assessed, or an aspect of the LLM design may have resulted in performance that differed from the expected question difficulty. We also considered whether the prompt structure could have been a confound. A previous psychometric evaluation of the SCS1 [112] included item trace plots as supplementary material [111]. These plots show the expected probability of selecting each multiple choice option for learners of varying knowledge levels. We used these plots to determine whether trends in incorrect LLM responses aligned with common student misconceptions.

4 RESULTS

4.1 RQ1: LLM Performance

Table 1 shows the results of person-fit statistics, with it and Figure 1 showing the knowledge level estimates for each LLM run with an acceptable fit. For the SCS1, we found that the 2PL model poorly fit the response patterns for Anthropic Claude v1.3 (zero and few shot), Llama 2 (7B) (zero and few shot), and Llama 2 (70B) (zero shot). The remaining models also had response patterns that reflected below average CS1 students, ranging from Llama 2 (7B) with few shot prompting ($\theta = -2.52 \pm 0.58$) to GPT-4 (0613) with zero shot prompting ($\theta = -0.33 \pm 0.22$).

Only five of the 20 LLM runs had responses that fit the 2PL model for the BDSI, with all other models being too noisy and poor of fits ($l_z < -2.0$). The LLMs with acceptable person-fit for the BDSI were all three Anthropic Claude models with few shot prompting and GPT-4 (0613) with zero and few shot prompting. These five instances produced responses that reflected CS2 students with data structures knowledge ranging from approximately average (GPT-4 (0613) with zero shot, $\theta = 0.12 \pm 0.44$ to above average (Anthropic Claude 2.0 with few shot, $\theta = 0.85 \pm 0.51$).

Table 1. Person fit statistics and knowledge estimates for LLMs with different prompting for the SCS1 and BDSI concept inventories. l_z is a person-fit statistic, with ** denoting unacceptable fit ($|l_z| > 2.0$). θ and standard error denote knowledge estimates of each LLM with an acceptable person fit.

		SCS1 (24 questions)			BDSI (11 questions)		
model name	prompting	l_z	θ	std. error	l_z	θ	std. error
claude-v1.3	zero shot	-2.31**	-	-	-4.05**	-	-
claude-v1.3	few shot	-2.79**	-	-	-0.76	0.66	0.48
claude-2.0	zero shot	-1.01	-1.06	0.32	-2.14**	-	-
claude-2.0	few shot	-1.13	-0.91	0.30	-1.59	0.46	0.46
claude-2.1	zero shot	-0.99	-1.01	0.31	-2.65**	-	-
claude-2.1	few shot	-0.36	-0.72	0.27	-0.31	0.85	0.51
gpt-3.5-turbo-0613	zero shot	-1.06	-1.45	0.39	-2.26**	-	-
gpt-3.5-turbo-0613	few shot	-0.26	-0.81	0.28	-2.26**	-	-
gpt-4-0613	zero shot	1.80	-0.33	0.22	-0.46	0.12	0.44
gpt-4-0613	few shot	0.62	-0.34	0.22	-0.31	0.13	0.44
llama-2-7b	zero shot	0.55	-2.52	0.58	-5.12**	-	-
llama-2-7b	few shot	-2.95**	-	-	-5.12**	-	-
llama-2-13b	zero shot	-2.34**	-	-	-4.35**	-	-
llama-2-13b	few shot	-0.71	-0.91	0.30	-4.91**	-	-
llama-2-70b	zero shot	-3.25**	-	-	-2.70**	-	-
llama-2-70b	few shot	-0.80	-1.11	0.33	-3.93**	-	-
mistral-7b-v0.1	zero shot	-0.42	-1.16	0.34	-4.50**	-	-
mistral-7b-v0.1	few shot	-0.58	-1.08	0.33	-2.26**	-	-
mixtral-8x7b-32kseqlen	zero shot	-1.34	-0.80	0.28	-3.09**	-	-
mixtral-8x7b-32kseqlen	few shot	-1.82	-0.61	0.25	-2.7**	-	-

Table 2. Number and percentage of invalid responses for LLM runs with zero and few shot in-context learning for each CI. Percentages calculated from 110 responses for each learning context for BDSI, and 240 responses for the SCS1.

	zero shot	few shot
BDSI	20 (18%)	0 (0%)
SCS1	46 (19%)	4 (2%)

Four runs produced responses that fit the 2PL models for both concept inventories: Anthropic Claude 2.0 and Anthropic Claude 2.1with few shot learning and GPT-4 (0613) with zero and few shot learning. In all four runs, response patterns for the SCS1 reflected below average CS1 knowledge ($\theta < 0$) but approximately average to above average knowledge of data structures ($\theta \geq 0$), typically considered a more advanced CS2 concept.

4.1.1 Validity Check: Number of Invalid Responses. LLM runs could still output invalid responses (e.g. a new line or word), which HELM would score as incorrect. We therefore attempted to correct for this using few shot in-context learning, as described in section 3.2.

Table 2 shows the frequency of invalid responses outputted from zero shot and few shot learning runs for each concept inventory. With zero-shot learning, we see that about 1 in 5 outputs are invalid. However, incorporating examples for few-shot learning resulted in no invalid responses for the BDSI and only 2% invalid responses for the SCS1 across all 10 LLMs. This suggests that for LLM runs for few-shot learning a valid answer is almost always outputted.

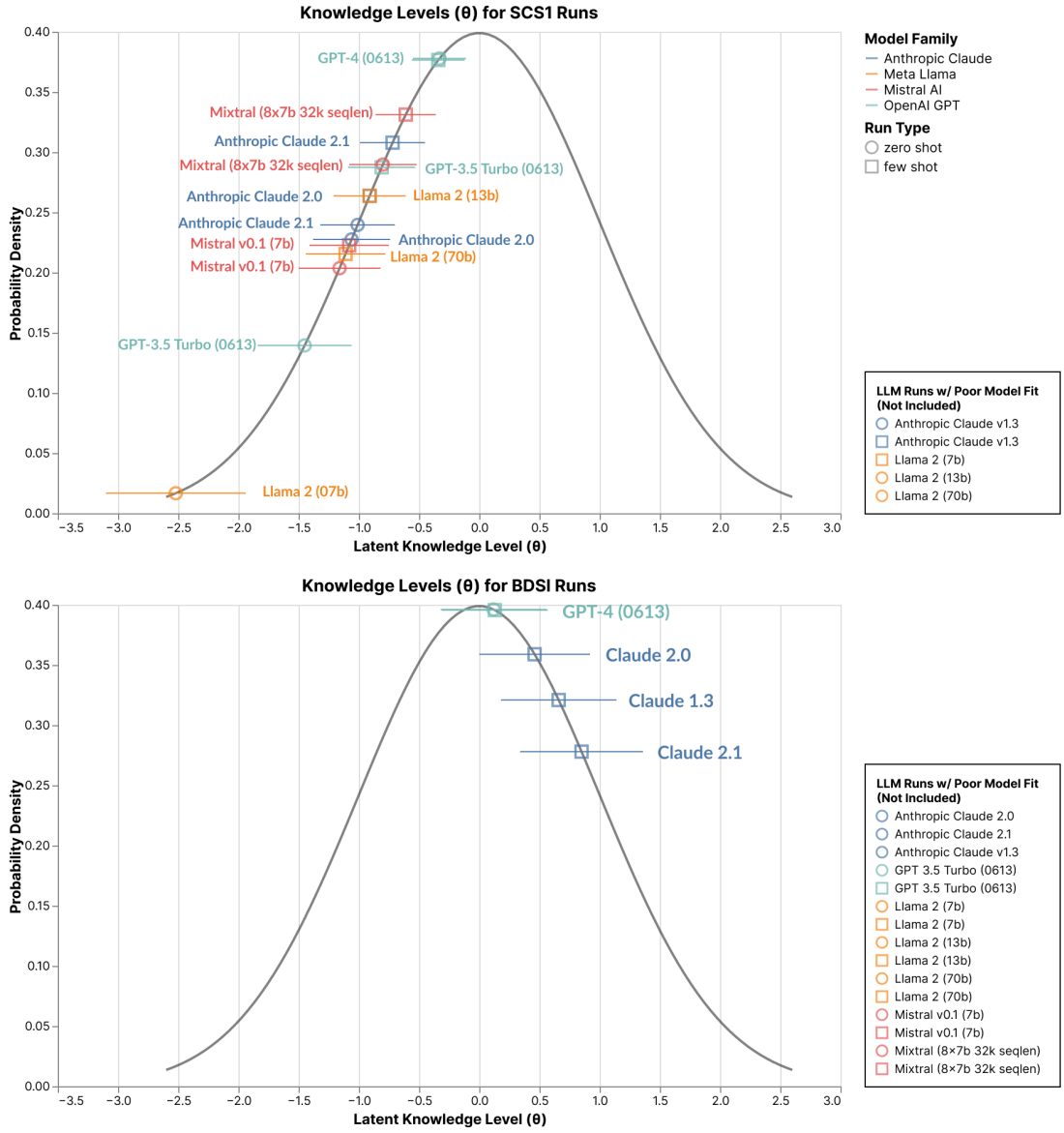

Fig. 1. Knowledge Estimates of LLM runs based on the the 2PL models for the SCS1 and BDS1 concept inventories. Models with unacceptable fit ($|l_z| > 2.0$) are noted in box on right and not plotted.

4.2 RQ2: Informal Expert Panel Review

To answer our second research question and identify potential explanations for deviations in LLM and expected student performance, we conducted an informal expert panel review with three authors, as described in Section 3.3.2. We

Table 3. Questions from the SCS1 and BDSI that we included in the expert panel review. Unusual items with high difficulty denote that LLMs performed better than expected student performance. Unusual items with low difficulty denote that LLMs performed worse than expected student performance. ** denotes that we designed a training item for in-context based on that item.

	SCS1	BDSI
High difficulty items which LLMs performed well on	Q15**, 7**, 26**	7**
Low difficulty items which LLMs performed poorly on	3, 19	4, 3**, 8

identified nine unusual items to include in our informal expert panel review, as shown in Table 3. In this section, we note key trends we identified [3].

For the expert panel review, we considered the information relating to item design (learning objective/concept, question type, number of responses, "select one" or "select all that apply," whether item includes code or images, correct answer, item and all responses), psychometric properties (difficulty and discrimination parameters from 2PL model, item trace plots for SCS1 items only), and LLM-related information (number of tokens in input prompt, whether training example was based on item, LLM run responses and correctness).

4.2.1 LLMs performed well on code-infill, poorly on nested conditionals. All three unusual high difficulty items for the SCS1 (Q15, 7, 26) were code infill/editing questions which provided a short code segment and required test-takers to fill in two or three gaps in the code with the correct code segments. They assessed the concepts of function return operators (Q15), while loops (Q7), and logical operators (Q26). Only 1-4% of CS1 students would be expected to get each question correct according to the extremely high item difficulties ($\alpha = 1.77 - 3.11$). However 20–45% of LLM runs got these questions correct. One potential explanation is that code infill/editing questions are a particular emphasis in LLM design [34], as evidenced by how GPT-4 (0613) and Llama 2 (7B) with few-shot learning correctly answered all three of these items. Another potential explanation is in how the question structure itself may have implicitly prompted the LLMs to answer these questions as a coding expert, such as how a prompt beginning with "you are an expert programmer" compared to "you are a novice programmer" can result in different outputs[106].

LLMs performed poorly on a low difficulty SCS1 item involving nested conditionals (SCS1 Q19). This item was relatively difficulty ($\alpha = 0.74$), with an expected 23% of CS1 students getting the item correct. LLMs did worse than this, with only 15% (3/20) of LLM runs getting this question correct: GPT-3.5 Turbo (0613) with few shot learning, GPT-4 (0613) with zero-shot learning, and Mistral v0.1 (7B) with zero-shot learning. Interestingly, both GPT-4 (0613) and Mistral v0.1 (7B) selected the correct response with zero-shot learning, but different incorrect responses with few-shot learning. We noticed the same distractor selected by 60% (12/20) of LLM runs, including three of the four most advanced models in their respective families with few shot learning: Anthropic Claude 2.1, Llama 2 (70B), and Mixtral (8x7B 32K seqlen). This distractor option reflected the erroneous execution of the outermost ELSE statement despite the corresponding IF block already being run. The item trace plot showed that this distractor was never the most commonly selected response option for CS1 students of any knowledge level. Therefore, the LLMs runs clustered towards a distractor that reflected a misconception that CS1 students did not often demonstrate.

[3]To maintain item security, we will not show actual items and will avoid naming correct and incorrect responses. We will mention exact question numbers to support knowledge building about these concept inventories across publications. This reporting practice is consistent with prior work (e.g. [41, 82, 86, 112])

4.2.2 LLMs clustering towards same distractors. The response pattern for the unusual low difficulty item SCS1 Q3 was one that aligned closely with student responses. This question asked test-takers to trace code with a while loop and select the correct output from five options. Based on the item difficulty ($\alpha = 0.27$), we would expect 39% of CS1 students to get this item correct. Only 25% (5/20) of scenarios outputted the correct response. Comparing LLM responses to the item trace plot revealed that most LLM runs (11/20) outputted the same distractor as the most commonly selected response amongst CS1 students with below average knowledge. These runs were all of smaller/older LLMs (Anthropic Claude v1.3, GPT-3.5 Turbo (0613), Mistral v0.1 (7B) with zero and few shot; Anthropic Claude 2.0 and Llama 2 (7B) with few shot; Llama 2 (13B) with zero shot) with the exception of Llama 2 (70B) with few shot. Only the most advanced models got the correct answer (Anthropic Claude 2.0 and GPT-4 (0613) with zero and few shot, Mixtral (8x7B 32K seqlen) with few shot). This pattern of LLM run outputs closely reflected expected CS1 student performance, with smaller LLMs selecting the same distractor as less knowledgeable CS1 students, and larger LLMs selecting the correct answer.

For BDSI Q4 (unusual low difficulty item), LLMs clustered towards a certain distractor that reflected the run time of typical mapping data structures. This item asked test-takers to consider the performance of implementing key-value interface with a linked list. This item was very easy ($\alpha = -2.71$), with over 99% of CS2 students expected to get it correct. However, only 50% of LLM runs the correct answer (include GPT-4 (0613) and Llama 2 (70B) for both zero and few shot and Anthropic Claude 2.1 with zero shot), with 25% selecting the same distractor (including Anthropic Claude 2.1 with few shot learning and Mixtral (8x7B 32K seqlen) with zero and few shot learning). That distractor reflected constant time efficiency. LLMs may have selected that distractor because the question described a mapping-like structure (like a hashmap), which typically have constant time lookup.

BDSI Q3 (unusual low difficulty item) also resulted in LLM runs clustering towards the same distractor with no clear explanation. We considered this question as requiring more creative problem solving because it asked students to identify the best experiment to determine whether a linked list was singly-linked or doubly-linked. It was a very easy item ($\alpha = -1.38$), with an expected 92% of CS2 students getting it correct. While Llama 2 (70B) and Mixtral (8x7B 32K seqlen) with zero-shot learning were the only two (10%) runs to output the correct response, the same two LLMs with few shot learning were amongst the 50% of runs that selected the same distractor. We could not identify a novice misconception that would lead to selecting that particular distractor. This was even more unusual because we had designed a training item based on the structure of this item, and the answer to the training item was the same as the distractor selected in half of LLM runs. Therefore, the training item could have affected model outputs, but we could not identify an exact explanation as to how. Another explanation for the poor LLM performance could be the extensive question description, which included an interface describing eight methods. This resulted in the item being the longest of all items we evaluated across both the BDSI and SCS1 (2801 tokens).

4.2.3 Modifications of item structure for integration in HELM may have affected difficulty. BDSI Q7 required test-takers to write a function to compute a property of a binary tree. The item was relatively difficult ($\alpha = 0.53$), with an expected 30% of CS2 students getting this question correct. LLMs performed well on this item with 55% (11/20) of LLM runs outputing the correct answer. A likely explanation was in how we had to change the answer structure to align with HELM capabilities.

The original item provided four potential options and required CS2 students to select all that applied. There was only one correct answer though. Because the HELM scenario we used could only allow one correct answer as an output, we effectively changed the structure of the question to prompt the LLMs to select exactly one option (further described in

Section 4.3.3). By changing the item from "select all that apply" to "select one," we reduced the number of valid answer options, potentially explaining why LLMs performed better than expected CS2 students.

4.3 RQ3: Challenges with Automating LLM Benchmarking

We answered our third research question by identifying challenges and opportunities with using automated benchmarking infrastructure in computing education research. Our goal was to use concept inventories as a benchmark and use the HELM infrastructure to automatically evaluate the performance of current (and hopefully future) LLMs against these CIs. In this section, we describe key challenges to using such a methodology:

4.3.1 Item security and benchmark leakage. Benchmark leakage (as described in Section 2.1.1) is a key concern to CIs because of risks of excessive item exposure. Questions in CIs do not change, so ensuring the security of these questions is crucial to the validity of interpreting and using their results. CI maintainers typically do not publicly post their instruments, instead granting access to their instruments on a case-by-case basis. We therefore assumed that CI items were not part of a training set for any LLMs. However, weak and shifting privacy policies typically enable LLM developers to use past prompts as future training [60, 91, 113, 115]. We describe our processes of gaining consent from CI maintainers and our failed attempts to limit benchmark leakage in Section 3.1. Using CIs as a benchmark opens a "Pandora's box" whereby we can assume that LLM developers will use CIs as future training data (as unlabeled data at least).

4.3.2 Faithful translation of CI items. As described in Section 3.2, HELM expects data to be passed in as a list of Instance objects, where an Instance represents a question and its associated answers. Typically, this data is prepared by first obtaining the requisite data set in JSON format, which streamlines the overall process (as well as allows users to take advantage of the many existing examples within HELM). As such, preparing CIs for HELM required us to convert them from their existing forms—generally human-readable PDFs—into JSON format. The difficulty here arose from making certain design decisions (see Section 3.2) which allowed us to represent CIs in such a format while ensuring we did not change the spirit of what the authors wished to convey. For example, the BDSI consisted of several questions involving graphs, which we needed to translate into a text representation both suitable for JSON and comprehensible to LLMs. Once these decisions were made, the translation was fairly straightforward, but time consuming.

4.3.3 Limited structure of items. We were limited to multiple-choice questions with single answers in our work, as it was unclear how to test for open-ended responses within HELM. Most directly, this limitation led to the exclusion of the Basic Recursion Concept Inventory [37] in our work, as it is a rare instance of a CI which asks open-ended free-response questions. Additionally, HELM does not currently support questions with multiple correct answers (i.e. "select all that apply" questions), which led to the exclusion of two such questions in the BDSI.

4.3.4 Development of training questions for HELM. While LLMs are popularly associated with more advanced models such as GPT-4 (0613), it is important to remember that not all models are quite so capable. Several models in HELM (such as Mistral v0.1 (7B) and Anthropic Anthropic Claude v1.3, in our work), require example question-answer pairs as few-shot training examples for in-context learning [105] in order to produce answers in the desired format. For most HELM examples, researchers can just take a subset of the existing data set as a training/test split. However, this approach was not suitable for CIs. Unlike other data sets, CIs are already small, and researchers will likely want to learn about LLM performance on all items. Using existing questions as few-shot training examples prevents those questions from being evaluated; thus, researchers must write additional example questions to mimic the structure of CI items. It

is worth noting, however, that these questions do not need to be high-quality assessment questions or even directly related to the topic of the CI; they simply need to match the input and output structure of the other questions.

4.3.5 Barriers to accessing and using LLMs within HELM. Firstly, the use of HELM required the preparation of data into a standardized machine-readable format and the use of a custom API, thereby requiring programming knowledge. Though LLM research can sometimes be done within user interfaces, such an approach does not scale for evaluating entire CIs and their individual items. Therefore, APIs must be used. Using a new type of data set for HELM requires researchers to prepare data in a standardized machine-readable format (see 3.2). This work contributes a new Scenario, called CIMCQA, to HELM, which future researchers may take advantage of. However, for other types of questions (e.g. free-response items), researchers will need to author their own Scenarios if they cannot adapt existing one. Furthermore, there are many subtleties within HELM itself that are difficult to navigate but essential to obtaining results. Our research was only made possible because an author was a member of HELM's research engineering team.

Finally, one of the more difficult challenges is obtaining access to all the available models in HELM. LLMs have barriers to access. Most are accessible through closed or limited APIs which users typically must pay for. Open-access LLMs often require owning or renting a CPU or paying for a commercial model interface platform to host the model. These constraints also apply to HELM. HELM users must have their own API keys for every single model they wish to use, a fairly limiting restriction considering the total number of models. These challenges to LLM access stand in the way of using benchmarking infrastructure to support replication of benchmark evaluations on future LLMs.

5 DISCUSSION: UNDERSTANDING OF LLM PERFORMANCE FOR MORE VALID ASSESSMENTS

In this paper, we explored the feasibility of using HELM, an automated benchmarking framework, as a methodology to support more reproducible, replicable, and rigorous empirical studies related to large language models (LLMs) in computing education. We evaluated the performance of 10 LLMs with zero and few-shot learning on two CS concept inventories (CIs): the SCS1 [82] to assess introductory computing knowledge, and the BDSI [86] to assess basic data structures knowledge. We found that most LLM runs produced SCS1 responses that aligned with below average introductory computing (CS1) students and BDSI responses that could not be modeled with a 2PL model. In an informal expert review comparing LLM performance to item-level psychometric properties, we found that LLMs performed well on code infil questions, but poorly on questions involving nested conditionals, runtime analysis, and longer question descriptions. We also identified challenges to our methodology involving CI item security, item translation, and barriers to LLM access. Through an empirical investigation that compares LLM and expected student performance using psychometric data and describes challenges to using HELM, this paper contributes as a feasibility study into the use of automated benchmarking infrastructure as a methodological innovation to computing education research.

In this section, we interpret our findings as they relate to the roles of LLMs in the iterative process of assessment design in computing education. We specifically consider the implications of using automated benchmarking infrastructure to investigate the rapidly evolving intersections of LLM capabilities, computing concepts, and assessment design.

5.1 Limitations and Threats to Validity

One interpretation of our study is that it suffers from limitations and threats to validity that invalidate our findings. This is of particular concern as we investigate the feasibility of a new methodology.

Our study suffers from convenience sampling of LLMs to include. We included 10 LLMs from 4 different organizations because we had access to their APIs. While these LLMs are popular and widely used at the time of our studies and prior

evaluations of LLMs typically consider fewer LLMs (e.g. [29, 56, 66, 98]), our findings are limited to the LLMs in this set. While they enable us to make hypotheses about how other LLMs might perform on the SCS1 and BDSI, we rely on future work to conduct hypothesis-driven studies that could yield more definitive claims. Furthermore, the HELM infrastructure makes evaluation of these CI items on additional LLMs more reliable, thus supporting replication of this study.

A threat to validity relates to the choices we made while translating CI items (particularly code and images) into the text passed into HELM. While we cannot make the example prompts we passed in public to maintain some level of item security, we attempted to be transparent in our design decisions in Section 3.2. Nevertheless, the possibility remains that some design decisions caused LLMs to interpret a CI item differently from how a student may interpret the original item, thus confounding our results. One example of this is how we changed a "select all that apply" item to a "select one" item to make it compatible with HELM infrastructure, as described in Section 4.2.3.

Another threat to validity is in our modeling of LLM responses as independent test-takers. Item Response Theory assumes that test-takers are providing responses independently of each other [19]. We make this assumption while recognizing that all 10 LLMs were created by four organizations. Furthermore, we attempt to model LLM runs as test-takers and measure their knowledge levels, θ. We checked for adequate fit to the psychometric models prior to reporting knowledge levels. We also used few-shot learning to reduce the number of invalid (and therefore incorrect) LLM responses to almost zero. Section 4.1 fully describes these checks that we conducted to ensure more valid comparison of LLM responses to expected student responses.

We also acknowledge that our informal expert review panel might have led to potentially biased results. We aim for this work to act as a pilot that sets the stage for future work with a larger panel and formally established methodology [73].

A final threat to validity is that benchmark leakage will invalidate future automated benchmarks. We should expect that LLMs may incorporate the CIs used for the benchmarks into their training data. This is currently an unavoidable risk, however, since few LLMs provide the facility to prevent leakage, and benchmarking cannot be done without passing in the data. Future work can investigate the use of techniques to mitigate and detect benchmark leakage [60, 91, 115].

5.2 Automating Closed-Ended Evaluations for Reproducibility and Replicability

Another interpretation of our findings is that the use of automated benchmarking infrastructure supports more reproducible and replicable evaluations of LLM performance on tasks related to computing education. As stated in Section 2.1, it is easier to automate more closed-ended tasks with clear measures of correctness, such as multiple-choice questions in CIs. While benchmarks are usually shared publicly, this should not occur to ensure the item security of CI items.

Before a CI can actually be passed into an LLM, it must be translated effectively into a format that a language model can interpret. It is well documented that prompting LLMs in certain ways can change the output [87, 106], and this extends to the CI questions themselves, because they become part of the prompt. While we documented our design decisions to aid replicability, we do not suggest our design decisions were ideal. The possibility remains that an alternative translation may have made greater sense to LLMs, or better enable them to mimic actual students. We identify this as a potential area of future work.

Our use of HELM helped address a key confound to evaluating LLM performance in assessments: determining correctness. Mahon et al. [68] identified how a majority of ChatGPT's incorrect responses to multiple choice questions involved ChatGPT outputting correct explanations, but incorrect or invalid outputs. Our use of HELM's few shot

in-context learning functionality reduced the number of invalid responses to 0-2% (Table 2), thereby almost eliminating the confound of LLMs producing an invalid response. Furthermore, sensitivity to prompt structure is a likely explanation for LLMs outputting incorrect responses despite correct explanations [49, 68]. Frameworks such as HELM can help address this by enabling more consistent and reusable structures of input prompts of LLMs. In our study, we developed the CIMCQA Scenario to do this (see Section 3.2), which future work can interrogate, utilize, and/or adapt. This transparent and more consistent prompt structuring can support more reproducible evaluations. Furthermore, HELM's continuous integration of new LLMs drastically reduces the barriers to future work that replicates previous evaluations on new LLMs, a key challenge to LLM research in computing education moving forward [87].

Taking a broader perspective, accessing and running more LLMs with HELM is perhaps the most pressing research challenge. More ambitious future work could involve extending this methodology to benchmark and evaluate all CIs with validity evidence against all common and modern LLMs in HELM. Though HELM provides the framework needed to carry out this work, it does not by default provide the monetary resources, as users must provide possess independent access for any model they desire to use. With the large number of LLMs available in HELM and at least 14 computing CIs with validity evidence [3, 41], the amount of resources needed is potentially best suited for a team of collaborative researchers.

A key implication of automated benchmarking infrastructure for reproducible and replicable computing education research is the investigation of hypotheses at the intersection of LLM capabilities, item design, and computing concepts. For example, our informal expert panel review (Section 4.2.1) identified that LLMs performed poorly on a low difficulty SCS1 question that CS1 students performed well on. This item involved tracing pseudocode which contained a nested conditional. We generated multiple explanations as to why LLM performance deviated from expected student performance, including aspects of the item design (LLMs may struggle to parse the pseudocode) or computing concepts being assessed (LLMs may struggle with nested conditionals). A computing education researcher or practitioner could feasibly investigate these hypotheses by designing items to explore each one (e.g. a pool of equivalent items in the pseudocode and other programming languages to determine LLM sensitivity to programming language, and a pool of different items assessing nested conditionals in the same programming language to determine LLM sensitivity to a computing concept). They could then use automated benchmarking infrastructure such as HELM to effectively control for confounding variables (e.g. prompt structure, determining correctness, etc.) when evaluating performance of different LLMs, thus supporting reproducibility. Crucially, they could also control for these confounds when evaluating new LLMs, thus supporting replicability.

5.3 Mixed-Methods Evaluations for Rigor

A final interpretation of our findings is that applying mixed-methods to connect LLM outputs with additional information such as psychometric evidence provides more rigorous investigations into the intersection of LLM capabilities, assessment design, and computing concepts.

For LLM runs with response patterns that fit the CI's 2PL models, we found that LLMs reflected knowledge of below average introductory CS knowledge but of above average data structures knowledge (see Section 4.1). We would not expect this pattern with any students because introductory CS knowledge is taught before and is typically considered pre-requisite knowledge to data structures. Future work can investigate whether this pattern is consistent and why this may be the case.

Existing psychometric data enabled our informal expert panel review to compare LLM responses to expected student responses. The use of concept inventories as benchmarks with existing validity evidence enabled more rigorous

evaluations. Furthermore, because concept inventories are specifically designed to identify student misconceptions, it was easier to pinpoint what exact topics LLMs struggled with when evaluating our results. This is a key difference between our benchmarking work as compared with prior work, which often used summative assessments from courses [96] or standardized assessments which may have validity evidence, but lack the structure and transparency of concept inventories to enable identification of misconceptions for specific concepts [49, 68]. In particular, we used item trace plots from Xie et al. [112] to consider whether LLMs and CS1 students of different knowledge levels selected similar or different distractors for the SCS1. This served as a key signal to identify whether LLM outputs reflected common student misconceptions (clustering towards similar distractors). When LLMs clustered towards uncommon distractors, we developed hypotheses related to item design, LLM design, and computing concepts, as described in Section 4.2. Mixed-methods future work could further explore these hypotheses to inform the design of computing pedagogy and assessment that are more resilient to changing LLM capabilities. Such future work could follow a similar mixed-methods approach by designing new items to benchmark, evaluating them with students and/or LLMs (using automated benchmarking infrastructure), and then conducting qualitative follow-up investigations with domain experts and students to better understand patterns. Future work could also investigate student perspectives on unusual items. Cognitive Interviews [27] with students in the target population could provide a more clear understanding of students' thought processes when answering questions, as well as new hypotheses on why LLMs perform differently than students on given items.

A key implication of more rigorous evaluations with automated benchmarking infrastructure is the opportunity for LLM-aided assessment design. Prior work in computing education has found that LLMs perform well on many programming tasks [1, 57, 65, 81, 87]. This is in part because LLM developers train their models on vast repositories of openly available code (e.g. GitHub) and have financial incentive to support programming tasks (e.g. code infill [34]). Therefore, LLM advancements will likely make the design of computing assessments particularly more challenging. Automated benchmarking infrastructure may be able to support assessment designers by simulating students to gauge item difficulty and/or using psychometric evidence to iteratively design more LLM-resilient items. Future work may also investigate the potential for LLMs to support automating aspects of item generation [47, 95], such as by generating informative distractors for multiple choice items [55, 69] or developing parallel items with similar difficulty [47].

6 CONCLUSION

In this paper, we described our process in efficiently and systematically evaluating LLM performance on computing education assessments with validity evidence. It is our hope that our work can contribute to a deeper understanding of the impact of LLMs on assessing students, as well as set the foundation for future work that can inform instructors in designing assessments that are more resilient against completion by LLMs.

Given the growing waves of LLMs and Generative AI tools, we want to acknowledge that advancements of LLMs do not necessarily translate to equitable benefits to students [12, 32, 67, 108]; assessments are not perfect or all-encompassing measures of learning [4, 75, 76]; and not all facets of computing are well-represented in AI tools [68, 87]. Therefore, we hope that computing education can leverage automated benchmarking infrastructure to encourage more cross-disciplinary, human-centered, and enduring investigations at the intersections of LLM capabilities, assessment design, and computing concepts in computing education.

ACKNOWLEDGMENTS

This material is based upon work supported by the Stanford Accelerator for Learning, Institute for Human-Centered Artificial Intelligence, Center for Research on Foundation Models, McCoy Family Center for Ethics in Society, and research credits provided by OpenAI. We thank Prof. Leif Wenar, Dr. Anne Newman, Dr. Shannon Sylvie Abelson, Dr. Moya Mapps, Dr. Ann Thresher, Prof. Ting-An Lin, Dr. Veronica Rivera, Dr. Jonathan Vandenburgh, and Dr. Daniel Webber of the McCoy Family Center for Ethics in Society for their cross-disciplinary insights and careful feedback across multiple iterations of this paper. We would also like to thank Dr. Sayamindu Dasgupta, the first author's Ph.D. advisor, for guidance and advice regarding the framing of this paper.

Documentation on how to use HELM can be found here: https://crfm-helm.readthedocs.io/en/latest/. The CIMCQAScenario we developed is available here: https://github.com/Murtz5253/helm/blob/main/src/helm/benchmark/scenarios/ci_mcqa_scenario.py. To preserve item security of concept inventories, we cannot share CI items. To access the concept inventories, please contact corresponding authors for Parker et al. [82] and Porter et al. [86]. Finally, we include all supplemental material to this paper (including the full document of design decisions taken to integrate CIs into HELM as well as our few-shot training items for HELM) at this public Github repository: https://github.com/Murtz5253/csed-benchmark-supplemental/tree/main

REFERENCES

[1] Vibhor Agarwal, Nakul Thureja, Madhav Krishan Garg, Sahiti Dharmavaram, Meghna, and Dhruv Kumar. 2024. "Which LLM should I use?": Evaluating LLMs for tasks performed by Undergraduate Computer Science Students in India. (Jan. 2024). arXiv:2402.01687 [cs.CY]

[2] Garima Agrawal, Kuntal Pal, Yuli Deng, Huan Liu, and Ying-Chih Chen. 2024. CyberQ: Generating Questions and Answers for Cybersecurity Education Using Knowledge Graph-Augmented LLMs. *Proceedings of the AAAI Conference on Artificial Intelligence* 38, 21 (Mar. 2024), 23164–23172. https://doi.org/10.1609/aaai.v38i21.30362

[3] Murtaza Ali, Sourojit Ghosh, Prerna Rao, Raveena Dhegaskar, Sophia Jawort, Alix Medler, Mengqi Shi, and Sayamindu Dasgupta. 2023. Taking Stock of Concept Inventories in Computing Education: A Systematic Literature Review. In *Proceedings of the 2023 ACM Conference on International Computing Education Research - Volume 1* (Chicago, IL, USA) *(ICER '23, Vol. 1)*. Association for Computing Machinery, New York, NY, USA, 397–415. https://doi.org/10.1145/3568813.3600120

[4] Mary J Allen and Wendy M Yen. 2001. *Introduction to Measurement Theory*. Waveland Press.

[5] Ryan S Baker. 2016. Stupid Tutoring Systems, Intelligent Humans. *International Journal of Artificial Intelligence in Education* 26, 2 (June 2016), 600–614. https://doi.org/10.1007/s40593-016-0105-0

[6] Rishabh Balse, Bharath Valaboju, Shreya Singhal, Jayakrishnan Madathil Warriem, and Prajish Prasad. 2023. Investigating the Potential of GPT-3 in Providing Feedback for Programming Assessments. In *Proceedings of the 2023 Conference on Innovation and Technology in Computer Science Education V. 1* (Turku, Finland) *(ITiCSE 2023)*. Association for Computing Machinery, New York, NY, USA, 292–298. https://doi.org/10.1145/3587102.3588852

[7] Erin M Bardar, Edward E Prather, Kenneth Brecher, and Timothy F Slater. 2007. Development and validation of the light and spectroscopy concept inventory. *Astronomy Education Review* 5, 2 (2007), 103–113.

[8] Yoshua Bengio, Réjean Ducharme, and Pascal Vincent. 2000. A neural probabilistic language model. *Advances in neural information processing systems* 13 (2000).

[9] BIG-bench authors. 2023. Beyond the Imitation Game: Quantifying and extrapolating the capabilities of language models. *Transactions on Machine Learning Research* (2023).

[10] Paulo Blikstein and Sepi Hejazi Moghadam. 2019. Computing Education Literature Review and Voices from the Field. In *The Cambridge Handbook of Computing Education Research*. Cambridge University Press, 56–78. https://doi.org/10.1017/9781108654555.004

[11] Borhane Blili-Hamelin and Leif Hancox-Li. 2023. Making Intelligence: Ethical Values in IQ and ML Benchmarks. In *Proceedings of the 2023 ACM Conference on Fairness, Accountability, and Transparency* (Chicago, IL, USA) *(FAccT '23)*. Association for Computing Machinery, New York, NY, USA, 271–284. https://doi.org/10.1145/3593013.3593996

[12] Su Lin Blodgett and Michael Madaio. 2021. Risks of AI Foundation Models in Education. (Oct. 2021). arXiv:2110.10024 [cs.CY]

[13] Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, Erik Brynjolfsson, Shyamal Buch, Dallas Card, Rodrigo Castellon, Niladri Chatterji, Annie Chen, Kathleen Creel, Jared Quincy Davis, Dora Demszky, Chris Donahue, Moussa Doumbouya, Esin Durmus, Stefano Ermon, John Etchemendy, Kawin Ethayarajh, Li Fei-Fei, Chelsea Finn, Trevor Gale, Lauren Gillespie, Karan Goel, Noah Goodman, Shelby Grossman, Neel Guha, Tatsunori Hashimoto, Peter Henderson, John Hewitt, Daniel E Ho, Jenny Hong, Kyle Hsu, Jing Huang, Thomas Icard, Saahil Jain, Dan Jurafsky, Pratyusha Kalluri, Siddharth

Karamcheti, Geoff Keeling, Fereshte Khani, Omar Khattab, Pang Wei Koh, Mark Krass, Ranjay Krishna, Rohith Kuditipudi, Ananya Kumar, Faisal Ladhak, Mina Lee, Tony Lee, Jure Leskovec, Isabelle Levent, Xiang Lisa Li, Xuechen Li, Tengyu Ma, Ali Malik, Christopher D Manning, Suvir Mirchandani, Eric Mitchell, Zanele Munyikwa, Suraj Nair, Avanika Narayan, Deepak Narayanan, Ben Newman, Allen Nie, Juan Carlos Niebles, Hamed Nilforoshan, Julian Nyarko, Giray Ogut, Laurel Orr, Isabel Papadimitriou, Joon Sung Park, Chris Piech, Eva Portelance, Christopher Potts, Aditi Raghunathan, Rob Reich, Hongyu Ren, Frieda Rong, Yusuf Roohani, Camilo Ruiz, Jack Ryan, Christopher Ré, Dorsa Sadigh, Shiori Sagawa, Keshav Santhanam, Andy Shih, Krishnan Srinivasan, Alex Tamkin, Rohan Taori, Armin W Thomas, Florian Tramèr, Rose E Wang, William Wang, Bohan Wu, Jiajun Wu, Yuhuai Wu, Sang Michael Xie, Michihiro Yasunaga, Jiaxuan You, Matei Zaharia, Michael Zhang, Tianyi Zhang, Xikun Zhang, Yuhui Zhang, Lucia Zheng, Kaitlyn Zhou, and Percy Liang. 2021. On the Opportunities and Risks of Foundation Models. (Aug. 2021). arXiv:2108.07258 [cs.LG]

[14] Neil C C Brown, Eva Marinus, and Aleata Hubbard Cheuoua. 2022. Launching Registered Report Replications in Computer Science Education Research. In *Proceedings of the 2022 ACM Conference on International Computing Education Research - Volume 1* (Lugano and Virtual Event, Switzerland) *(ICER '22)*. Association for Computing Machinery, New York, NY, USA, 309–322. https://doi.org/10.1145/3501385.3543971

[15] Ricardo Caceffo, Pablo Frank-Bolton, Renan Souza, and Rodolfo Azevedo. 2019. Identifying and Validating Java Misconceptions Toward a CS1 Concept Inventory. In *Proceedings of the 2019 ACM Conference on Innovation and Technology in Computer Science Education* (Aberdeen, Scotland Uk) *(ITiCSE '19)*. Association for Computing Machinery, New York, NY, USA, 23–29. https://doi.org/10.1145/3304221.3319771

[16] 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. 2023. A Survey on Evaluation of Large Language Models. (July 2023). arXiv:2307.03109 [cs.CL]

[17] Will Crichton, Gavin Gray, and Shriram Krishnamurthi. 2023. A Grounded Conceptual Model for Ownership Types in Rust. *Proc. ACM Program. Lang.* 7, OOPSLA2 (Oct. 2023), 1224–1252. https://doi.org/10.1145/3622841

[18] C H Crouch and E Mazur. 2001. Peer Instruction: Ten years of experience and results. *Am. J. Phys.* 69 (Aug. 2001), 970–977. https://doi.org/10.1119/1.1374249

[19] R J De Ayala. 2009. *The theory and practice of item response theory.* Guilford Press, New York.

[20] Adrian de Freitas, Joel Coffman, Michelle de Freitas, Justin Wilson, and Troy Weingart. 2023. FalconCode: A Multiyear Dataset of Python Code Samples from an Introductory Computer Science Course. In *Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 1* (Toronto ON, Canada,) *(SIGCSE 2023)*. Association for Computing Machinery, New York, NY, USA, 938–944. https://doi.org/10.1145/3545945.3569822

[21] Mostafa Dehghani, Yi Tay, Alexey A Gritsenko, Zhe Zhao, Neil Houlsby, Fernando Diaz, Donald Metzler, and Oriol Vinyals. 2021. The Benchmark Lottery. (July 2021). arXiv:2107.07002 [cs.LG]

[22] Andre Del Carpio Gutierrez, Paul Denny, and Andrew Luxton-Reilly. 2024. Evaluating Automatically Generated Contextualised Programming Exercises. In *Proceedings of the 55th ACM Technical Symposium on Computer Science Education V. 1* (Portland, Oregon, USA) *(SIGCSE 2024)*. Association for Computing Machinery, New York, NY, USA, 289–295. https://doi.org/10.1145/3626252.3630863

[23] Jwala Dhamala, Tony Sun, Varun Kumar, Satyapriya Krishna, Yada Pruksachatkun, Kai-Wei Chang, and Rahul Gupta. 2021. BOLD: Dataset and Metrics for Measuring Biases in Open-Ended Language Generation. In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency* (Virtual Event, Canada) *(FAccT '21)*. Association for Computing Machinery, New York, NY, USA, 862–872. https://doi.org/10.1145/3442188.3445924

[24] Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Zhiyong Wu, Baobao Chang, Xu Sun, Jingjing Xu, Lei Li, and Zhifang Sui. 2023. A Survey on In-context Learning. arXiv:2301.00234 [cs.CL]

[25] Fritz Drasgow, Michael V Levine, and Esther A Williams. 1985. Appropriateness measurement with polychotomous item response models and standardized indices. *Br. J. Math. Stat. Psychol.* 38, 1 (May 1985), 67–86. https://doi.org/10.1111/j.2044-8317.1985.tb00817.x

[26] Jerome Epstein. 2007. Development and validation of the Calculus Concept Inventory. In *Proceedings of the ninth international conference on mathematics education in a global community*, Vol. 9. Citeseer, 165–170.

[27] Karl Anders Ericsson and Herbert Alexander Simon. 1993. *Protocol Analysis: Verbal Reports as Data Revised Edition.* The MIT Press.

[28] Hans Eysenck and Steven Rose. 1979. Race, intelligence and education. *New community* 7, 2 (June 1979), 278–283. https://doi.org/10.1080/1369183X.1979.9975576

[29] James Finnie-Ansley, Paul Denny, Andrew Luxton-Reilly, Eddie Antonio Santos, James Prather, and Brett A Becker. 2023. My AI wants to know if this will be on the exam: Testing OpenAI's codex on CS2 programming exercises. In *Australasian Computing Education Conference* (Melbourne VIC Australia). ACM, New York, NY, USA. https://doi.org/10.1145/3576123.3576134

[30] James Finnie-Ansley, Paul Denny, Andrew Luxton-Reilly, Eddie Antonio Santos, James Prather, and Brett A. Becker. 2023. My AI Wants to Know if This Will Be on the Exam: Testing OpenAI's Codex on CS2 Programming Exercises. In *Proceedings of the 25th Australasian Computing Education Conference* (Melbourne, VIC, Australia) *(ACE '23)*. Association for Computing Machinery, New York, NY, USA, 97–104. https://doi.org/10.1145/3576123.3576134

[31] Melissa J Gillis and Andrew T Jacobs. 2019. *Introduction to Women's and Gender Studies: An Interdisciplinary Approach.* Oxford University Press.

[32] Global Future Council on Artificial Intelligence for Humanity. 2022. *A Blueprint for Equity and Inclusion in Artificial Intelligence.* Technical Report. World Economic Forum.

[33] Ken Goldman, Paul Gross, Cinda Heeren, Geoffrey Herman, Lisa Kaczmarczyk, Michael C. Loui, and Craig Zilles. 2008. Identifying important and difficult concepts in introductory computing courses using a delphi process. *SIGCSE Bull.* 40, 1 (mar 2008), 256–260. https://doi.org/10.1145/

1352322.1352226

[34] Linyuan Gong, Sida Wang, Mostafa Elhoushi, and Alvin Cheung. 2024. Evaluation of LLMs on Syntax-Aware Code Fill-in-the-Middle Tasks. (March 2024). arXiv:2403.04814 [cs.CL]

[35] S Hall, F G Abrantes, Hanwen Zhu, Grace A Sodunke, Aleksandar Shtedritski, and Hannah Rose Kirk. 2023. VisoGender: A dataset for benchmarking gender bias in image-text pronoun resolution. *Adv. Neural Inf. Process. Syst.* abs/2306.12424 (June 2023). https://doi.org/10.48550/arXiv.2306.12424

[36] Ibrahim Abou Halloun and David Hestenes. 1985. The initial knowledge state of college physics students. *Am. J. Phys.* 53, 11 (Nov. 1985), 1043–1055. https://doi.org/10.1119/1.14030

[37] Sally Hamouda, Stephen H Edwards, Hicham G Elmongui, Jeremy V Ernst, and Clifford A Shaffer. 2017. A basic recursion concept inventory. *Computer Science Education* 27, 2 (2017), 121–148.

[38] Sarah Heckman, Jeffrey C Carver, Mark Sherriff, and Ahmed Al-zubidy. 2021. A Systematic Literature Review of Empiricism and Norms of Reporting in Computing Education Research Literature. *ACM Trans. Comput. Educ.* 22, 1 (Oct. 2021), 1–46. https://doi.org/10.1145/3470652

[39] Dan Hendrycks, Steven Basart, Saurav Kadavath, Mantas Mazeika, Akul Arora, Ethan Guo, Collin Burns, Samir Puranik, Horace He, Dawn Song, et al. 2021. Measuring coding challenge competence with apps. *arXiv preprint arXiv:2105.09938* (2021).

[40] 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. (March 2021). arXiv:2103.03874 [cs.LG]

[41] Geoffrey L Herman, Shan Huang, Peter A Peterson, Linda Oliva, Enis Golaszewski, and Alan T Sherman. 2023. Psychometric Evaluation of the Cybersecurity Curriculum Assessment. In *Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 1* (Toronto ON, Canada) *(SIGCSE 2023)*. Association for Computing Machinery, New York, NY, USA, 228–234. https://doi.org/10.1145/3545945.3569762

[42] Geoffrey L. Herman, Michael C. Loui, and Craig Zilles. 2010. Creating the digital logic concept inventory. In *Proceedings of the 41st ACM Technical Symposium on Computer Science Education* (Milwaukee, Wisconsin, USA) *(SIGCSE '10)*. Association for Computing Machinery, New York, NY, USA, 102–106. https://doi.org/10.1145/1734263.1734298

[43] Matthew Hertz. 2010. What do "CS1" and "CS2" mean? investigating differences in the early courses. In *Proceedings of the 41st ACM Technical Symposium on Computer Science Education* (Milwaukee, Wisconsin, USA) *(SIGCSE '10)*. Association for Computing Machinery, New York, NY, USA, 199–203. https://doi.org/10.1145/1734263.1734335

[44] David Hestenes, Malcolm Wells, Gregg Swackhamer, and Others. 1992. Force concept inventory. *Phys. Teach.* 30, 3 (1992), 141–158.

[45] Charles L Hulin, Fritz Drasgow, and Charles K Parsons. 1983. *Item Response Theory: Application to Psychological Measurement.* Dow Jones-Irwin.

[46] Frederick Jelinek. 1998. *Statistical methods for speech recognition.* MIT press.

[47] Hong Jiao and Robert W Lissitz. 2020. *Application of Artificial Intelligence to Assessment.* IAP.

[48] Di Jin, Eileen Pan, Nassim Oufattole, Wei-Hung Weng, Hanyi Fang, and Peter Szolovits. 2021. What Disease Does This Patient Have? A Large-Scale Open Domain Question Answering Dataset from Medical Exams. *NATO Adv. Sci. Inst. Ser. E Appl. Sci.* 11, 14 (July 2021), 6421. https://doi.org/10.3390/app11146421

[49] Ishika Joshi, Ritvik Budhiraja, Harshal Dev, Jahnvi Kadia, Mohammad Osama Ataullah, Sayan Mitra, Harshal D. Akolekar, and Dhruv Kumar. 2024. ChatGPT in the Classroom: An Analysis of Its Strengths and Weaknesses for Solving Undergraduate Computer Science Questions. In *Proceedings of the 55th ACM Technical Symposium on Computer Science Education V. 1* (Portland, OR, USA) *(SIGCSE 2024)*. Association for Computing Machinery, New York, NY, USA, 625–631. https://doi.org/10.1145/3626252.3630803

[50] Lisa C. Kaczmarczyk, Elizabeth R. Petrick, J. Philip East, and Geoffrey L. Herman. 2010. Identifying student misconceptions of programming. In *Proceedings of the 41st ACM Technical Symposium on Computer Science Education* (Milwaukee, Wisconsin, USA) *(SIGCSE '10)*. Association for Computing Machinery, New York, NY, USA, 107–111. https://doi.org/10.1145/1734263.1734299

[51] Michael T Kane. 2013. Validating the Interpretations and Uses of Test Scores. *Journal of Educational Measurement* 50, 1 (March 2013), 1–73. https://doi.org/10.1111/jedm.12000

[52] Masahiro Kaneko, Danushka Bollegala, Naoaki Okazaki, and Timothy Baldwin. 2024. Evaluating Gender Bias in Large Language Models via Chain-of-Thought Prompting. (Jan. 2024). arXiv:2401.15585 [cs.CL]

[53] Kuba Karpierz and Steven A. Wolfman. 2014. Misconceptions and concept inventory questions for binary search trees and hash tables. In *Proceedings of the 45th ACM Technical Symposium on Computer Science Education* (Atlanta, Georgia, USA) *(SIGCSE '14)*. Association for Computing Machinery, New York, NY, USA, 109–114. https://doi.org/10.1145/2538862.2538902

[54] Theodoros A Kyriazos and Anastasios Stalikas. 2018. Applied psychometrics: The steps of scale development and standardization process. *Psychology* 09, 11 (2018), 2531–2560. https://doi.org/10.4236/psych.2018.911145

[55] Hollis Lai, Mark J Gierl, Claire Touchie, Debra Pugh, André-Philippe Boulais, and André De Champlain. 2016. Using Automatic Item Generation to Improve the Quality of MCQ Distractors. *Teach. Learn. Med.* 28, 2 (2016), 166–173. https://doi.org/10.1080/10401334.2016.1146608

[56] Juho Leinonen, Paul Denny, Stephen MacNeil, Sami Sarsa, Seth Bernstein, Joanne Kim, Andrew Tran, and Arto Hellas. 2023. Comparing Code Explanations Created by Students and Large Language Models. (April 2023). arXiv:2304.03938 [cs.CY]

[57] 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 the 54th ACM Technical Symposium on Computer Science Education V. 1* (Toronto ON, Canada) *(SIGCSE 2023)*. Association for Computing Machinery, New York, NY, USA, 563–569. https://doi.org/10.1145/3545945.3569770

[58] Colleen M Lewis, Huda Khayrallah, and Amy Tsai. 2013. Mining data from the AP CS a exam: patterns, non-patterns, and replication failure. In *Proceedings of the ninth annual international ACM conference on International computing education research* (San Diego, San California, USA) *(ICER*

'13). Association for Computing Machinery, New York, NY, USA, 115–122. https://doi.org/10.1145/2493394.2493415

[59] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461* (2019).

[60] Changmao Li and Jeffrey Flanigan. 2023. Task Contamination: Language Models May Not Be Few-Shot Anymore. (Dec. 2023). arXiv:2312.16337 [cs.CL]

[61] Percy Liang, Rishi Bommasani, Tony Lee, Dimitris Tsipras, Dilara Soylu, Michihiro Yasunaga, Yian Zhang, Deepak Narayanan, Yuhuai Wu, Ananya Kumar, Benjamin Newman, Binhang Yuan, Bobby Yan, Ce Zhang, Christian Cosgrove, Christopher D Manning, Christopher Ré, Diana Acosta-Navas, Drew A Hudson, Eric Zelikman, Esin Durmus, Faisal Ladhak, Frieda Rong, Hongyu Ren, Huaxiu Yao, Jue Wang, Keshav Santhanam, Laurel Orr, Lucia Zheng, Mert Yuksekgonul, Mirac Suzgun, Nathan Kim, Neel Guha, Niladri Chatterji, Omar Khattab, Peter Henderson, Qian Huang, Ryan Chi, Sang Michael Xie, Shibani Santurkar, Surya Ganguli, Tatsunori Hashimoto, Thomas Icard, Tianyi Zhang, Vishrav Chaudhary, William Wang, Xuechen Li, Yifan Mai, Yuhui Zhang, and Yuta Koreeda. 2022. Holistic Evaluation of Language Models. (Nov. 2022). arXiv:2211.09110 [cs.CL]

[62] Percy Liang, Yifan Mai, Josselin Somerville, Farzaan Kaiyom, Tony Lee, and Rishi Bommasani. 2023. HELM Lite: Lightweight and Broad Capabilities Evaluation. https://crfm.stanford.edu/2023/12/19/helm-lite.html. Accessed: 2024-3-2.

[63] Julie Libarkin. 2008. Concept Inventories in Higher Education Science. In *National Research Council Promising Practices in Undergraduate STEM Education Workshop*, Vol. 13. 14.

[64] Julie C Libarkin and Steven W Anderson. 2005. Assessment of learning in entry-level geoscience courses: Results from the Geoscience Concept Inventory. *Journal of Geoscience Education* 53, 4 (2005), 394–401.

[65] Stephen MacNeil, Andrew Tran, Arto Hellas, Joanne Kim, Sami Sarsa, Paul Denny, Seth Bernstein, and Juho Leinonen. 2023. Experiences from Using Code Explanations Generated by Large Language Models in a Web Software Development E-Book. In *Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 1* (Toronto ON, Canada) *(SIGCSE 2023)*. Association for Computing Machinery, New York, NY, USA, 931–937. https://doi.org/10.1145/3545945.3569785

[66] Stephen MacNeil, Andrew Tran, Dan Mogil, Seth Bernstein, Erin Ross, and Ziheng Huang. 2022. Generating Diverse Code Explanations using the GPT-3 Large Language Model. In *Proceedings of the 2022 ACM Conference on International Computing Education Research - Volume 2* (Lugano and Virtual Event, Switzerland) *(ICER '22, Vol. 2)*. Association for Computing Machinery, New York, NY, USA, 37–39. https://doi.org/10.1145/3501709.3544280

[67] Michael Madaio, Su Lin Blodgett, Elijah Mayfield, and Ezekiel Dixon-Román. 2021. Beyond "fairness:" structural (in)justice lenses on AI for education. (May 2021). arXiv:2105.08847 [cs.CY]

[68] Joyce Mahon, Brian Mac Namee, and Brett A. Becker. 2023. No More Pencils No More Books: Capabilities of Generative AI on Irish and UK Computer Science School Leaving Examinations. In *Proceedings of the 2023 Conference on United Kingdom & Ireland Computing Education Research* (Swansea, Wales Uk,) *(UKICER '23)*. Association for Computing Machinery, New York, NY, USA, Article 2, 7 pages. https://doi.org/10.1145/3610969.3610982

[69] Wojciech Malec. 2024. Investigating the quality of AI-generated distractors for a multiple-choice vocabulary test. https://doi.org/10.5220/0012762400003693

[70] Lauri Malmi, Judy Sheard, Päivi Kinnunen, Simon, and Jane Sinclair. 2019. Computing Education Theories: What Are They and How Are They Used?. In *Proceedings of the 2019 ACM Conference on International Computing Education Research* (Toronto ON, Canada) *(ICER '19)*. Association for Computing Machinery, New York, NY, USA, 187–197. https://doi.org/10.1145/3291279.3339409

[71] Nestor Maslej, Loredana Fattorini, Erik Brynjolfsson, John Etchemendy, Katrina Ligett, Terah Lyons, James Manyika, Helen Ngo, Juan Carlos Niebles, Vanessa Parli, Yoav Shoham, Russell Wald, Jack Clark, and Raymond Perrault. 2023. *The AI Index 2023 Annual Report*. Technical Report. Stanford University.

[72] J Nathan Matias, Sayamindu Dasgupta, and Benjamin Mako Hill. 2016. Skill Progression in Scratch Revisited. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*. ACM, New York, NY, USA, 1486–1490. https://doi.org/10.1145/2858036.2858349

[73] Daniel McCaffrey, Jodi Casabianca, Kathryn Ricker-Pedley, René Lawless, and Cathy Wendler. 2021. *Best Practices for Constructed-Response Scoring*. Technical Report. Educational Testing Services.

[74] Monica M McGill, Tom McKlin, and Errol Kaylor. 2019. Defining What Empirically Works Best: Dynamic Generation of Meta-Analysis for Computer Science Education. In *Proceedings of the 2019 ACM Conference on International Computing Education Research* (Toronto ON, Canada) *(ICER '19)*. Association for Computing Machinery, New York, NY, USA, 199–207. https://doi.org/10.1145/3291279.3339401

[75] Samuel Messick. 1993. Validity. In *Educational Measurement. Third Edition. American Council on Education Series on Higher Education*. Oryx Press, 4041 North Central at Indian School Road, Phoenix, AZ 85012-3397., 13–103.

[76] Samuel Messick. 1995. Validity of psychological assessment: Validation of inferences from persons' responses and performances as scientific inquiry into score meaning. *The American psychologist* 50, 9 (Sept. 1995), 741–749. https://doi.org/10.1037/0003-066X.50.9.741

[77] Briana B Morrison, Adrienne Decker, and Lauren E Margulieux. 2016. Learning Loops: A Replication Study Illuminates Impact of HS Courses. In *Proceedings of the 2016 ACM Conference on International Computing Education Research* (Melbourne, VIC, Australia) *(ICER '16)*. Association for Computing Machinery, New York, NY, USA, 221–230. https://doi.org/10.1145/2960310.2960330

[78] Douglas R Mulford and William R Robinson. 2002. An inventory for alternate conceptions among first-semester general chemistry students. *Journal of chemical education* 79, 6 (2002), 739.

[79] Moin Nadeem, Anna Bethke, and Siva Reddy. 2020. StereoSet: Measuring stereotypical bias in pretrained language models. (April 2020). arXiv:2004.09456 [cs.CL]

[80] Greg L Nelson and Amy J Ko. 2018. On Use of Theory in Computing Education Research. In *Proceedings of the 2018 ACM Conference on International Computing Education Research*. ACM.

[81] Eng Lieh Ouh, Benjamin Kok Siew Gan, Kyong Jin Shim, and Swavek Wlodkowski. 2023. ChatGPT, Can You Generate Solutions for my Coding Exercises? An Evaluation on its Effectiveness in an undergraduate Java Programming Course. In *Proceedings of the 2023 Conference on Innovation and Technology in Computer Science Education V. 1* (Turku, Finland) *(ITiCSE 2023)*. Association for Computing Machinery, New York, NY, USA, 54–60. https://doi.org/10.1145/3587102.3588794

[82] Miranda C Parker, Mark Guzdial, and Shelly Engleman. 2016. Replication, Validation, and Use of a Language Independent CS1 Knowledge Assessment. In *Proceedings of the 2016 ACM Conference on International Computing Education Research (ICER '16)*. ACM, New York, NY, USA, 93–101. https://doi.org/10.1145/2960310.2960316

[83] Alicia Parrish, Angelica Chen, Nikita Nangia, Vishakh Padmakumar, Jason Phang, Jana Thompson, Phu Mon Htut, and Samuel Bowman. 2022. BBQ: A hand-built bias benchmark for question answering. In *Findings of the Association for Computational Linguistics: ACL 2022*. Association for Computational Linguistics, Dublin, Ireland, 2086–2105. https://doi.org/10.18653/v1/2022.findings-acl.165

[84] Baolin Peng, Xiujun Li, Lihong Li, Jianfeng Gao, Asli Celikyilmaz, Sungjin Lee, and Kam-Fai Wong. 2017. Composite Task-Completion Dialogue Policy Learning via Hierarchical Deep Reinforcement Learning. (2017). https://doi.org/10.18653/v1/d17-1237

[85] Gustavo Pinto, Isadora Cardoso-Pereira, Danilo Monteiro, Danilo Lucena, Alberto Souza, and Kiev Gama. 2023. Large Language Models for Education: Grading Open-Ended Questions Using ChatGPT. In *Proceedings of the XXXVII Brazilian Symposium on Software Engineering* (Campo Grande, Brazil) *(SBES '23)*. Association for Computing Machinery, New York, NY, USA, 293–302. https://doi.org/10.1145/3613372.3614197

[86] Leo Porter, Daniel Zingaro, Soohyun Nam Liao, Cynthia Taylor, Kevin C Webb, Cynthia Lee, and Michael Clancy. 2019. BDSI: A Validated Concept Inventory for Basic Data Structures. In *Proceedings of the 2019 ACM Conference on International Computing Education Research* (Toronto ON, Canada) *(ICER '19)*. Association for Computing Machinery, New York, NY, USA, 111–119. https://doi.org/10.1145/3291279.3339404

[87] James Prather, Paul Denny, Juho Leinonen, Brett A Becker, Ibrahim Albluwi, Michelle Craig, Hieke Keuning, Natalie Kiesler, Tobias Kohn, Andrew Luxton-Reilly, Stephen MacNeil, Andrew Petersen, Raymond Pettit, Brent N Reeves, and Jaromir Savelka. 2023. The Robots Are Here: Navigating the Generative AI Revolution in Computing Education. In *Proceedings of the 2023 Working Group Reports on Innovation and Technology in Computer Science Education* (Turku, Finland) *(ITiCSE-WGR '23)*. Association for Computing Machinery, New York, NY, USA, 108–159. https://doi.org/10.1145/3623762.3633499

[88] Arif Rachmatullah, Bita Akram, Danielle Boulden, Bradford Mott, Kristy Boyer, James Lester, and Eric Wiebe. 2020. Development and validation of the middle grades computer science concept inventory (MG-CSCI) assessment. *EURASIA Journal of Mathematics, Science and Technology Education* 16, 5 (2020), em1841.

[89] Inioluwa Deborah Raji, Emily M Bender, Amandalynne Paullada, Emily Denton, and Alex Hanna. 2021. AI and the Everything in the Whole Wide World Benchmark. (Nov. 2021). arXiv:2111.15366 [cs.LG]

[90] Sam Saarinen, Shriram Krishnamurthi, Kathi Fisler, and Preston Tunnell Wilson. 2019. Harnessing the Wisdom of the Classes: Classsourcing and Machine Learning for Assessment Instrument Generation. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education* (Minneapolis, MN, USA) *(SIGCSE '19)*. Association for Computing Machinery, New York, NY, USA, 606–612. https://doi.org/10.1145/3287324.3287504

[91] Oscar Sainz, Jon Ander Campos, Iker García-Ferrero, Julen Etxaniz, Oier Lopez de Lacalle, and Eneko Agirre. 2023. NLP Evaluation in trouble: On the Need to Measure LLM Data Contamination for each Benchmark. (Oct. 2023). arXiv:2310.18018 [cs.CL]

[92] Eddie Antonio Santos, Prajish Prasad, and Brett A. Becker. 2023. Always Provide Context: The Effects of Code Context on Programming Error Message Enhancement. In *Proceedings of the ACM Conference on Global Computing Education Vol 1* (Hyderabad, India) *(CompEd 2023)*. Association for Computing Machinery, New York, NY, USA, 147–153. https://doi.org/10.1145/3576882.3617909

[93] Sami Sarsa, Paul Denny, Arto Hellas, and Juho Leinonen. 2022. Automatic Generation of Programming Exercises and Code Explanations Using Large Language Models. In *Proceedings of the 2022 ACM Conference on International Computing Education Research - Volume 1* (Lugano and Virtual Event, Switzerland) *(ICER '22, Vol. 1)*. Association for Computing Machinery, New York, NY, USA, 27–43. https://doi.org/10.1145/3501385.3543957

[94] Sami Sarsa, Paul Denny, Arto Hellas, and Juho Leinonen. 2022. Automatic Generation of Programming Exercises and Code Explanations Using Large Language Models. In *Proceedings of the 2022 ACM Conference on International Computing Education Research - Volume 1* (Lugano and Virtual Event, Switzerland) *(ICER '22)*. Association for Computing Machinery, New York, NY, USA, 27–43. https://doi.org/10.1145/3501385.3543957

[95] Andreas Säuberli and Simon Clematide. 2024. Automatic Generation and Evaluation of Reading Comprehension Test Items with Large Language Models. (April 2024). arXiv:2404.07720 [cs.CL]

[96] Jaromir Savelka, Arav Agarwal, Marshall An, Chris Bogart, and Majd Sakr. 2023. Thrilled by Your Progress! Large Language Models (GPT-4) No Longer Struggle to Pass Assessments in Higher Education Programming Courses. In *Proceedings of the 2023 ACM Conference on International Computing Education Research - Volume 1* (Chicago, IL, USA,) *(ICER '23)*. Association for Computing Machinery, New York, NY, USA, 78–92. https://doi.org/10.1145/3568813.3600142

[97] Jaromir Savelka, Arav Agarwal, Christopher Bogart, and Majd Sakr. 2023. Large Language Models (GPT) Struggle to Answer Multiple-Choice Questions about Code. (March 2023). arXiv:2303.08033 [cs.CL]

[98] Jaromir Savelka, Arav Agarwal, Christopher Bogart, Yifan Song, and Majd Sakr. 2023. Can Generative Pre-trained Transformers (GPT) Pass Assessments in Higher Education Programming Courses? (March 2023). arXiv:2303.09325 [cs.AI]

[99] Stanford Center for Research on Foundation Models. 2022. Ecosystem Graphs for Foundation Models. https://crfm.stanford.edu/ecosystem-graphs/index.html?mode=table. Accessed: 2024-3-12.

[100] Andrew Taylor, Alexandra Vassar, Jake Renzella, and Hammond Pearce. 2024. dcc –help: Transforming the Role of the Compiler by Generating Context-Aware Error Explanations with Large Language Models. In *Proceedings of the 55th ACM Technical Symposium on Computer Science Education V. 1* (Portland, Oregon, USA) *(SIGCSE 2024)*. Association for Computing Machinery, New York, NY, USA, 1314–1320. https://doi.org/10.1145/3626252.3630822

[101] Cynthia Taylor, Daniel Zingaro, Leo Porter, Kevin C Webb, Cynthia Bailey Lee, and Mike Clancy. 2014. Computer science concept inventories: past and future. *Computer Science Education* 24, 4 (2014), 253–276.

[102] Allison Elliott Tew and Mark Guzdial. 2011. The FCS1: a language independent assessment of CS1 knowledge. In *Proceedings of the 42nd ACM Technical Symposium on Computer Science Education* (Dallas, TX, USA) *(SIGCSE '11)*. Association for Computing Machinery, New York, NY, USA, 111–116. https://doi.org/10.1145/1953163.1953200

[103] The National Science Foundation and The Institute of Education Sciences. 2018. *Companion Guidelines on Replication & Reproducibility in Education Research*. Technical Report. NSF and IES.

[104] Jan Vahrenhold and Wolfgang Paul. 2014. Developing and validating test items for first-year computer science courses. *Computer Science Education* 24, 4 (2014), 304–333. https://doi.org/10.1080/08993408.2014.970782 arXiv:https://doi.org/10.1080/08993408.2014.970782

[105] Yaqing Wang, Quanming Yao, James T. Kwok, and Lionel M. Ni. 2020. Generalizing from a Few Examples: A Survey on Few-shot Learning. *ACM Comput. Surv.* 53, 3, Article 63 (jun 2020), 34 pages. https://doi.org/10.1145/3386252

[106] Jules White, Quchen Fu, Sam Hays, Michael Sandborn, Carlos Olea, Henry Gilbert, Ashraf Elnashar, Jesse Spencer-Smith, and Douglas C. Schmidt. 2023. A Prompt Pattern Catalog to Enhance Prompt Engineering with ChatGPT. arXiv:2302.11382 [cs.SE]

[107] R. Paul Wiegand, Anthony Bucci, Amruth N. Kumar, Jennifer L. Albert, and Alessio Gaspar. 2016. A Data-Driven Analysis of Informatively Hard Concepts in Introductory Programming. In *Proceedings of the 47th ACM Technical Symposium on Computing Science Education* (Memphis, Tennessee, USA) *(SIGCSE '16)*. Association for Computing Machinery, New York, NY, USA, 370–375. https://doi.org/10.1145/2839509.2844629

[108] Ben Williamson. 2024. AI in education is a public problem. https://codeactsineducation.wordpress.com/2024/02/22/ai-in-education-is-a-public-problem/. Accessed: 2024-5-30.

[109] Tianyu Wu, Shizhu He, Jingping Liu, Siqi Sun, Kang Liu, Qing-Long Han, and Yang Tang. 2023. A Brief Overview of ChatGPT: The History, Status Quo and Potential Future Development. *IEEE/CAA Journal of Automatica Sinica* 10, 5 (2023), 1122–1136. https://doi.org/10.1109/JAS.2023.123618

[110] Changrong Xiao, Sean Xin Xu, Kunpeng Zhang, Yufang Wang, and Lei Xia. 2023. Evaluating Reading Comprehension Exercises Generated by LLMs: A Showcase of ChatGPT in Education Applications. In *Proceedings of the 18th Workshop on Innovative Use of NLP for Building Educational Applications (BEA 2023)*, Ekaterina Kochmar, Jill Burstein, Andrea Horbach, Ronja Laarmann-Quante, Nitin Madnani, Anaïs Tack, Victoria Yaneva, Zheng Yuan, and Torsten Zesch (Eds.). Association for Computational Linguistics, Toronto, Canada, 610–625. https://doi.org/10.18653/v1/2023.bea-1.52

[111] Benjamin Xie. 2019. Supplementary Info for "An Item Response Theory Evaluation of a Language-Independent CS1 Knowledge Assessment" (Xie et al. SIGCSE 2019). https://github.com/codeandcognition/archive-2019sigcse-xie. Accessed: 2024-1-15.

[112] Benjamin Xie, Matthew J Davidson, Min Li, and Amy J Ko. 2019. An Item Response Theory Evaluation of a Language-Independent CS1 Knowledge Assessment. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education* (Minneapolis, MN, USA) *(SIGCSE '19)*. ACM, 699–705. https://doi.org/10.1145/3287324.3287370

[113] Yifan Yao, Jinhao Duan, Kaidi Xu, Yuanfang Cai, Zhibo Sun, and Yue Zhang. 2024. A survey on Large Language Model (LLM) security and privacy: The Good, The Bad, and The Ugly. *High-Confidence Computing* (March 2024), 100211. https://doi.org/10.1016/j.hcc.2024.100211

[114] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, Yifan Du, Chen Yang, Yushuo Chen, Zhipeng Chen, Jinhao Jiang, Ruiyang Ren, Yifan Li, Xinyu Tang, Zikang Liu, Peiyu Liu, Jian-Yun Nie, and Ji-Rong Wen. 2023. A Survey of Large Language Models. (2023). arXiv:2303.18223 [cs.CL]

[115] Kun Zhou, Yutao Zhu, Zhipeng Chen, Wentong Chen, Wayne Xin Zhao, Xu Chen, Yankai Lin, Ji-Rong Wen, and Jiawei Han. 2023. Don't Make Your LLM an Evaluation Benchmark Cheater. (Nov. 2023). arXiv:2311.01964 [cs.CL]