Measuring Algorithm Understanding in Humans and AI

Mirabel Reid (mirabel.reid@tu-darmstadt.de)[1], Khushi Sharma[2] & Anand Singh[1]
[1]School of Computer Science, Georgia Institute of Technology
[2]Yale University

Abstract

Understanding algorithms is a central learning goal for those in computational fields and is frequently used as a cognitive benchmark for contemporary large language models (LLMs). We present a hierarchy that characterizes algorithmic understanding in terms of observable performance on both concrete and abstract tasks to juxtapose human and LLM performance and draw conclusions about the latter's understanding of the subject space. We evaluate the hierarchy in a controlled assessment study comparing 43 university students with eight LLMs across five canonical algorithms. The resulting performance profiles reveal that LLMs often excel at tasks dependent on concept retrieval while failing on tasks requiring computational example construction. These differences mirror distinctions in rote versus meaningful learning and suggest that algorithm understanding in humans and LLMs may rely on different underlying resources.

Keywords: Large Language Models; Algorithmic Reasoning; Understanding; Human-AI Cognition

Introduction

Motivation

Large language models (LLMs) are increasingly trusted to act autonomously in complex environments. When comparing models, accuracy on set benchmarks is often treated as a definitive measure of ability. More nuanced evaluations of subject-specific knowledge have remained under-assessed, constraining detailed comparisons of the computational abilities of LLMs to those of humans beyond question-specific accuracy. In particular, *understanding*, a crucial metric we use to assign responsibility in humans, has seen few efforts to formally assess and quantify in LLMs.

In this work, we refine and evaluate a precise, testable definition of understanding as applied to algorithms. Algorithmic thinking is a cornerstone of several primary applications of LLMs, such as coding assistance and tutoring. Since understanding algorithms is also a key pedagogical goal in computer science, this is a unique opportunity to directly compare human and LLM cognition. We establish a metric to reflect the differences in computational— and ultimately cognitive— abilities that LLMs exhibit in comparison to our own.

Our approach builds on the framework proposed by Reid and Vempala (2025), who devised a hierarchy of understanding that could be applied to any entity with memory and computational powers. The hierarchy was additionally validated via a study on human subjects and successive generations of GPT, showing that GPT understood at a higher level.

We extend this line of inquiry in several respects. First, we refine the hierarchy of algorithmic understanding and ground the levels using precise definitions. We evaluate a diverse set of LLMs against a variety of algorithms, which enables a broader exploration of performance variations. Further, we relate performance across algorithmic thinking tasks to several properties of the task and algorithm, discussed further below.

Axes of Evaluation

Although our focus is algorithm understanding, the space of algorithmic tasks is broad, and relative performance across humans and LLMs depends on specific characteristics of the algorithm and the task. We analyze performance across three axes: input type, abstraction level, and understanding level.

Input Type. Input type refers to the structure of the object manipulated in the algorithm, such as a number, array, matrix, or graph. Inputs may have internal structure or a physical interpretation, which affects the types of manipulations that can be performed.

Abstraction Level. Tasks related to algorithms can range in abstraction from concrete, requiring specific, instantiated manipulations, to abstract, requiring reasoning on higher-level concepts and categorizations.

Understanding Level. Tasks can also vary independently in the depth of understanding required to complete them. We categorize algorithmic tasks using a refinement of the hierarchy proposed by Reid and Vempala (2025), who defined successive levels of algorithm understanding based on the types of tasks that an entity can consistently perform.

In this study, we develop a theory of how human and LLM ability to operate with algorithms varies across these axes. Rather than comparing task performance directly, we analyze the error patterns that shape each performance profile.

Related Work

Reasoning and Cognition in Language Models. The question of whether LLMs exhibit human-like cognitive abilities is a rapidly evolving area of research. Some investigations include metacognitive reasoning (Didolkar et al., 2024), creativity (Bellemare-Pepin et al., 2024; Zhao et al., 2025), theory of mind (Li et al., 2023), and cognitive biases (Koo et al., 2023; Macmillan-Scott & Musolesi, 2024). Understanding the parallels between human and LLM cognition can enable

In G. Boleda, I. Dautriche, M. Maldonado, C. Saldana and T. Torrent (Eds.), *Proceedings of the 48th Annual Conference of the Cognitive Science Society*

the use of AI to simulate humans and understand human behavior (Aher et al., 2023; Hagendorff et al., 2023).

Measurements of Understanding. Educational psychologists distinguish *deep* and *surface-level* learning, measuring how the depth of understanding of a subject evolves through the learning process (Beattie et al., 1997; Marton & Säljö, 1976). A classical framework for categorizing this progression is Bloom's taxonomy (Bloom et al., 1956; Mayer, 2002), which organizes learning objectives into six cognitive levels of increasing complexity. In the context of *algorithm* understanding, Perrenet et al. (2005), identified four levels of abstraction in algorithmic thinking in computer science students, varying from execution-level to problem-level reasoning. Formal definitions of understanding also arise in philosophy (Baumberger et al., 2016; Friedman, 1974; Wilkenfeld, 2013) and AI (Mahowald et al., 2024; Mitchell & Krakauer, 2023). We contribute via a formal, mathematical categorization of understanding that is specific to algorithms.

Preliminaries

We first define the central objects in our framework: the entity that exhibits understanding and the algorithm that is understood. These definitions establish a common formal language for discussing both human and artificial reasoners.

Definition 1. *Let $\mathcal{E}$ be a computational entity. Assume that our entity can (1) read and write to a bounded internal storage, and (2) perform a specified set of base computational operations.*

Definition 2. *Let $\mathcal{A}$ be an algorithm. The algorithm is defined to take in an input x and returns a result $f_{\mathcal{A}}(x)$. Assume that $\mathcal{A}$ can be expressed as a combination of control-flow operations and operations computable by the entity.*

The *execution path* of $\mathcal{A}$ on input x is the sequence of states visited during the execution. The *trace* of the execution modifies the path such that it contains the contents of the entity's internal storage (or tape) at each step.

A *trace property* (or simply *property*) of $\mathcal{A}$ is a boolean function over traces that indicates whether a run of $\mathcal{A}$ on an input x satisfies the property. A *subroutine* of $\mathcal{A}$ is an algorithm invoked by $\mathcal{A}$ that contributes to its output; i.e., $\mathcal{A}$ is composed of a set of subroutines and control-flow operations. Because most arbitrary Boolean functions are not semantically meaningful, we restrict our attention to *simple properties*.

Definition 3 (Simple Property)**.** *A property P is a simple property if (1) there exists a circuit or formula that returns whether the trace of execution on x satisfies P, and (2) there exists an algorithm G_P that, given a bit length k, returns an input x of length at most k satisfying P (or reports that none exists) in time $o(2^k)$ (i.e., without exhaustive search).*

Simple properties capture behavior that can be efficiently identified. We use this definition to ground the abstract levels of understanding to tractable, semantically meaningful tasks.

Hierarchy of Understanding

In this section, we refine the levels of understanding proposed by Reid and Vempala (2025). A *level* denotes a set of tasks that an entity can correctly perform. The levels are not strictly hierarchical, but they correspond roughly to increasing complexity for humans. The names of the levels are derived from the cognitive processes identified by Mayer (2002), adapted to the domain of algorithms. They are illustrated with example tasks in Figure 1, with task complexity increasing from the top to the bottom of the figure.

Let $S_{\mathcal{A}}$ be a string which compiles in a formal programming language and implements $\mathcal{A}$. We assume that $\mathcal{E}$ knows the programming language semantics, and syntactic mistakes are not treated as evidence of misunderstanding.

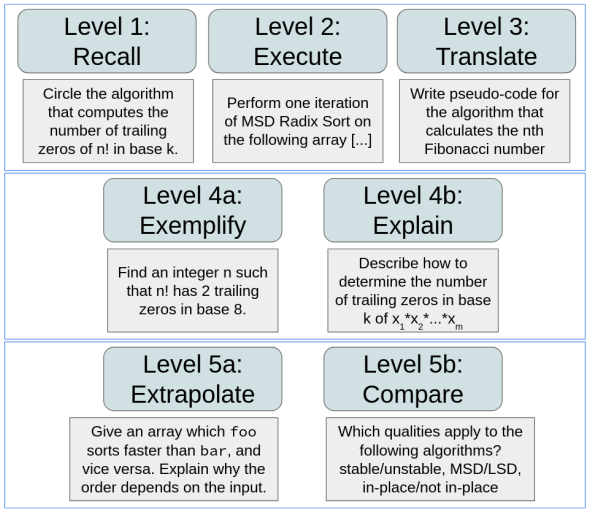

Figure 1: A diagram of the levels of understanding and the corresponding degree of abstraction. For each level, we include an example task from the Factorial, Radix Sort, or Dynamic Programming exam given in the study.

Shallow Understanding Levels. The first three understanding levels correspond to tasks that deal with simple manipulation and recognition and would be encountered with a surface-level exposure to the algorithm. Given a specific program $S_{\mathcal{A}}$, the entity can recognize the program (Level 1), execute the sequence of instructions it defines (Level 2), and recall the program, or translate it from a description to code (Level 3).

Definition (Level 1: Recognize)**.** *Given a set of candidate programs, the entity can identify the specific program $S_{\mathcal{A}}$ implementing $\mathcal{A}$.*

Definition (Level 2: Execute)**.** *Given an input, the entity can provide the sequence of operations performed by the program on that input and deliver the correct output.*

Definition (Level 3: Translate)**.** *Given a precise description of the function $\mathcal{A}$ computes, the entity can recall and output the string of a program implementing $\mathcal{A}$.*

The three levels capture the fundamental cognitive processes related to concrete algorithm understanding.

Deep Understanding Levels. At Level 4, the entity can reason about the components of the algorithm and their properties, and it's divided into two task types: *exemplify* and *explain*.

Definition (Level 4a: Exemplify)**.** *Given a simple property P, the entity can return an input satisfying the property within a time constraint.*

Definition (Level 4b: Explain)**.** *Given an audience, an entity can describe the steps of the program's execution, shorten that description dependent on what the audience knows, and use other natural language terms to convey the basis of the program to the audience.*

At Level 5, the entity can reason about perturbations of the algorithm or its input and determine their effect on the execution path. We again divide Level 5 into two task types: *extrapolate* and *compare*.

Definition (Level 5a: Extrapolate)**.** *Given an alternate algorithm on the same input space and a simple property, the entity can determine within a time constraint whether the property holds for the same inputs on both algorithms. If not, it can output a counterexample x where the value of the property differs.*

Definition (Level 5b: Compare)**.** *Given an alternate algorithm, the entity can describe a property that highlights the difference between the two algorithms while also being able to shorten the description of the alternate using comparisons to elements of the original.*

Levels 4 and 5 together capture reasoning over the conceptual space of algorithmic structure, properties, and transformation. These levels represent the boundary between understanding algorithms as a sequence of concrete actions and understanding them as objects that can be employed and designed to solve a problem.

Method

We constructed a comparative study to assess the proposed hierarchy and test how the above axes of evaluation affect algorithm understanding in humans and LLMs. We first lay out our main research questions and hypotheses, then describe the experimental procedures to assess them.

Research Questions

RQ1 (Understanding Level): *Does the proposed hierarchy capture differences in algorithm understanding?*
Hypothesis 1: We hypothesize that human performance will increase with algorithms training at higher understanding levels. Likewise, we expect LLMs optimized for reasoning to outperform non-reasoning models at higher understanding levels.
RQ2 (Input Type): *How does performance vary with the structure and complexity of the algorithm's input?*
Hypothesis 2: We hypothesize that LLM performance will decrease as the dimension or geometric complexity of the input increases (e.g., moving from arrays to graphs). Student performance will be less sensitive to the change in input structure.
RQ3 (Abstraction Level): *How does performance vary depending on the level of abstraction of the task?*
Hypothesis 3: We hypothesize that LLMs will perform worse on concrete computational tasks, while human performance will decrease with abstraction level.
RQ4 (Performance Profile): *Does the relative performance of LLMs on algorithmic reasoning tasks align with that of students?*
Hypothesis 4: We hypothesize that the relative difficulty of tasks will vary between students and LLMs. We expect that LLMs will perform comparatively better on tasks which leverage familiar patterns or prior knowledge and worse on tasks requiring concrete example construction.

Study Description

The study focused on five algorithms, listed in Table 1, with five tasks per algorithm varying across abstraction and understanding levels.

These algorithms were selected by the authors for wide variance in input type and problem difficulty for both models and humans. Through an iterative process, the algorithms and question sets were tweaked to be distinguishable from problems the LLMs and human subjects may have encountered in courses and online resources, while also embodying principles and control flows that are fundamental and familiar.

Human Study We established a human baseline via a study of students (n = 43; 4 undergraduate, 29 masters, and 10 Doctoral) at a top U.S. engineering university. Most participants were actively enrolled in computer science programs, with a few exceptions in mathematics and other related engineering majors. Most participants reported that they had previously taken algorithms courses, and fifteen participants had previously been a Teaching Assistant in an algorithms course. Since the algorithms we assess involve standard topics in undergraduate and graduate algorithms curricula, we treat students with prior coursework or teaching experience as expert participants.

The participants were randomly assigned three of five algorithms to be assessed on and were instructed to complete the two they were most comfortable with. This procedure was implemented to preserve coverage across algorithms while reducing noise due to unfamiliarity. The per-question response rate is shown in Figure 2. They were given 45 minutes to complete the exam. All scratch-work was collected. Participants received a $10 Amazon gift card for participation, with no additional performance-based incentives.

The study was classified as exempt by the Institutional Review Board, and informed consent was obtained from all participants. Responses were anonymized prior to analysis.

Table 1: List of algorithms used in human study.

Algorithm	Description	Input Type	Control Flow
Factorial	Compute the number of zeros in the base k representation of $n!$.	Integer	Iterative
Radix Sort	Sort an array of integers by passing through their base n digits.	Array	Recursive
Rotting Oranges	Track the spread of a property that travels along graph edges.	Graph	Search
Knight's Tour	Compute the number of ways a knight can reach a specified square on an $n \times n$ chessboard.	Integer	Dynamic Programming
Blocks-world	Find the shortest plan that transforms a tower of labeled blocks from the initial state to a goal state.	Array	Brute Force

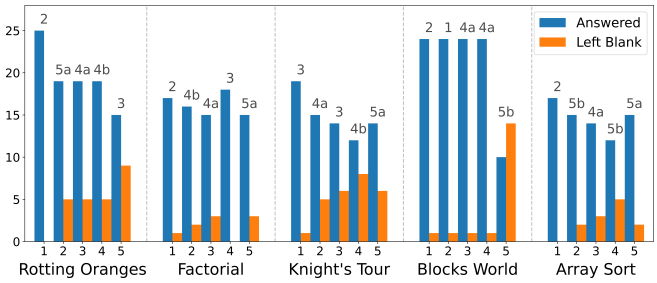

Figure 2: The number of participants who answered/left blank each question. The numbers above the bars indicate the understanding level targeted by each question.

LLM Study We evaluated eight different LLMs[1] with the same set of questions administered to the human participants. Models were classified as 'reasoning' if they used documented reasoning optimizations such as chain of thought. Each LLM was queried via an API using the default temperature settings. To ensure stability, the assessment was performed five times per model.

Scoring The responses (both human and LLM) were assessed on a scale of 0-2, with 2 for full credit, 1 for partial credit, and 0 for no credit. The responses were hand-graded by the authors using a common rubric. To prevent grading bias, the responses for each LLM were all graded together, and the name of the model which produced the response was masked. Blank responses were excluded from scoring; the rate of blank responses are indicated in Figure 2.

Results

RQ1 (Understanding Level). We validate our hierarchy of understanding by comparing the performance on each type of task between reasoning/non-reasoning models and and between students with varying algorithms experience. The results of this are shown in Figure 3. The performance of reasoning models was on average higher than that of models without documented reasoning optimizations (significant with $p < 0.05$). For students, the performance aligned with their experience with algorithms. Students who had taken a graduate algorithms course had an average per-question score of 1.37, while students who had not taken a graduate algorithms course averaged 1.07 (a statistically significant difference with $p < 0.05$).

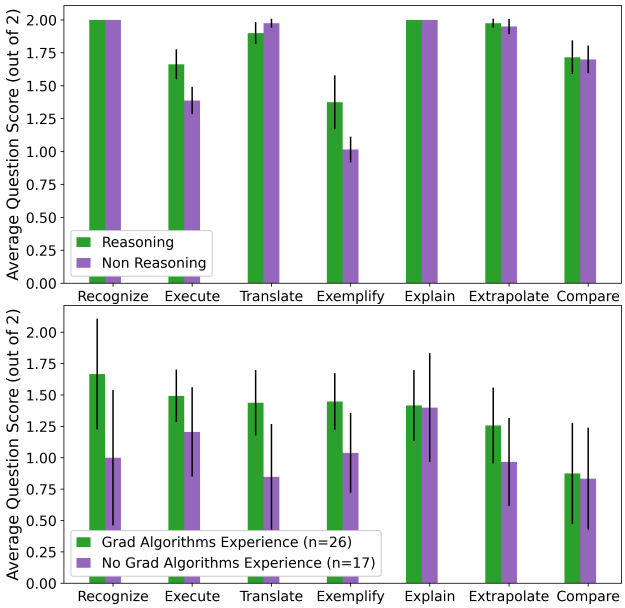

Figure 3: The performance on tasks at each understanding level, broken down by whether the model is optimized for reasoning (Top) and whether the student had taken a graduate-level algorithms course (Bottom).

The level of variation between human and LLM scores was highly dependent on the type of task that they were assessed on. LLMs outperformed humans on average on the Recall, Translate, Explain, Compare, and Extrapolate tasks (significant with $p < 0.05$). On the Execute and Exemplify tasks, the performance was commensurate and the difference in means was not statistically significant. Both of these categories of tasks were computational, leading to more opportunity for error in step-by-step calculations.

RQ2 (Input Type). The average human and LLM performance per algorithm is illustrated in Figure 4. While the questions between exams varied in difficulty, we calibrate this

[1]Reasoning Models used: Claude Opus 4.1 (2025-05-14), DeepSeek V3.1 Reasoner, GPT-5 (2025-08-07), o4-mini (2025-04-16). Non-Reasoning Models used: Claude Sonnet 4 (2025-05-14), DeepSeek V3.1 Chat, Gemini 2.5 Flash, GPT-4.1 (2025-04-14).

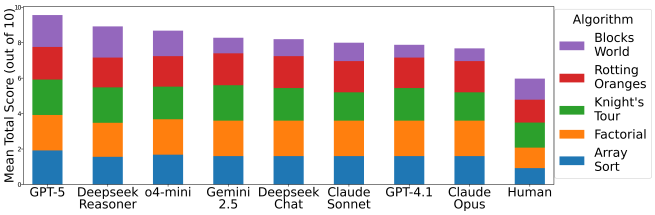

Figure 4: The average total score broken down by model. Human average is shown for comparison.

by examining the difference in scores between humans and LLMs. Large language models had the greatest gain over students on Factorial (numerical input) and Array Sort (array input) tasks , with a mean gain of 0.83/2 points and 0.72/2 points per question respectively. On the other hand, there was no statistically significant difference in scores between students and LLMs on the Blocks World exam. This supports the idea that LLMs perform better on lower-dimensional numerical tasks, while struggling on tasks with physical interpretations.

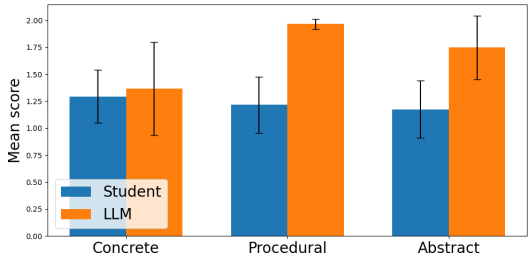

Figure 5: The average scores across student and language model responses, grouped by concrete, procedural, and abstract questions. Error bars show the 95% confidence interval.

RQ3 (Abstraction Level). The tasks were grouped into three categories reflecting increasing levels of abstraction in reasoning: direct calculation using specific, instantiated inputs (*concrete*), construction of specific generalizable instructions such as code (*procedural*), and reasoning over general properties and relationships (*abstract*). Figure 5 illustrates the average human and LLM scores in each of these categories. Human scores did not show a statistically significant difference between the categories; however, LLMs had a statistically significant performance gap between concrete tasks and the procedural and abstract categories.

RQ4 (Performance Profile). We examined how well different models' score distributions aligned with that of the student participants. This alignment was measured via Spearman rank correlation – a correlation coefficient, ranging between -1 and 1, that measures the linear correlation between two populations. The result of this is shown in Figure 6. Confidence intervals were computed by bootstrapping the student responses ($B = 2000$ resamples). Deepseek V3.1 had the greatest score alignment with students, while GPT-5 was the least aligned.

Out of the 25 questions posed to both students and LLMs, LLMs had a higher mean score on 20 of them. We examine the specific questions further in the qualitative section.

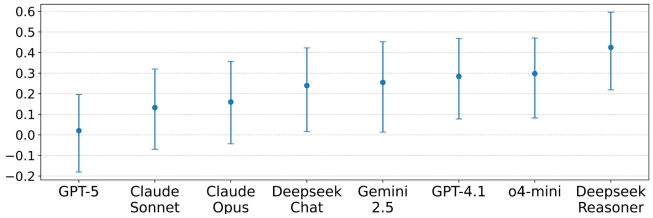

Figure 6: The Spearman rank correlation between the mean per-question performance for models and students. Error bars show the 95% confidence interval, computed via bootstrapping with random re-sampling of student answers.

Qualitative

The questions with the highest performance gap between LLMs and students highlight systemic differences in algorithm understanding. We find that LLMs are able to outperform humans when they are able to connect a question to a common concept or technique. On the other hand, students are more robust to subtle deviations from standard concepts, particularly on example-based reasoning.

LLMs recognize familiar algorithmic patterns. The question at which LLMs had the greatest performance gain over students was on the Rotting Oranges exam, a canonical problem used to assess competency in breadth-first search (BFS). We presented a variant framed as disease spread on a social network and asked how to extend a function `infect(v, graph)`, defined for a single initial infected vertex, to handle multiple initial vertices.

The intended solution uses *multi-source BFS*: creating a 'super source' connected to all the individuals who are ill on Day 1, and noting that the disease spreads for a single additional day. LLMs reliably identified this as a variant of a standard idea, and averaged a score of 1.85 on this question. Students averaged a score of only 0.58; many participants suggested calling infect(v, graph) for each v in the list and aggregating the results, an intuitive but incorrect strategy.

LLMs fail on example and counterexample questions. Students outperformed LLMs on two questions requiring the construction of specific examples. On one question, respondents were given a specific implementation of Radix Sort and were asked to provide an array that terminates after three iterations of the while loop. Students had an average score of 0.86, while LLMs scored only 0.25. The loop structure of the given algorithm was non-standard, so many LLM-generated responses implicitly assumed a standard loop condition and failed to correctly count the iterations. Students as a whole did not make this error.

A similar pattern appeared on a Blocks World question asking whether a transformation between any valid initial and goal states is always possible. Students had an average score of 1.46, while LLMs scored 0.975. Since the question took place in a non-standard Blocks World variation — in which

blocks can only be placed on a limited number of spaces — the correct answer is false. However, the statement would be true in the standard variant of Blocks World, and the models often responded with such. While some students got this wrong, many were able to reason that the goal is not always reachable if the number of stacks is limited to 1 or 2.

Summary. Both of the 'difficult' questions we noted above involve computing an example or counter-example fitting a specified property. We find that this pattern held throughout the study. As shown in Figure 5, LLMs performed worst on *concrete* problems such as the array sort question, and performed better on *procedural* and *abstract* questions, which involved writing pseudocode or explaining reasoning (as in the graph question). This aligns with hypothesis that LLMs excel on tasks drawing from prior knowledge but struggle with more straightforward, example-based questions.

Discussion

We found notable differences in performance patterns on algorithmic tasks between LLMs and humans. Human performance scaled with algorithmic training as predicted by conventional wisdom in educational psychology (see e.g. Mayer, 2002); participants who learned algorithms at only a rote level were not able to access all the cognitive processes accessible to more experienced students. However, while LLM performance was higher than students on average, the performance patterns were atypical; this reinforces our claim that overall algorithmic task performance in LLMs is not be comparable to human experts.

We note that we intentionally differentiated the abstraction level of the task from the depth of understanding it tests. While it is tempting to argue that abstract reasoning subsumes concrete, this relationship does not always hold. For example, an expert researcher may have a deep understanding of the ideas behind an algorithm, the proof of correctness, and its relation to other problems; however, they may not necessarily be able to reproduce the fine details of its implementation. Computer science curricula typically build a foundation of concrete understanding before introducing more abstract modes of analysis (Perrenet et al., 2005). However, we did not observe significant performance variation in students between abstraction levels (see Figure 5). LLMs, on the other hand, do not undergo a staged learning process based on abstraction, and their performance on concrete tasks was significantly lower.

We also replicate the pattern noted by Reid and Vempala (2025): LLMs had a significant gap in performance between 'translate' (code production) tasks and 'execute' tasks, while the opposite trend was observed in humans. One possible explanation lies in architectural differences between humans and language models. In humans, formal language and mathematical processing occur in interacting but distinct areas, while in contrast, LLMs encode all forms of reasoning within the context of next-token prediction (Mahowald et al., 2024). This structural difference suggests that human-like reasoning patterns on explicit computational tasks may require augmenting LLMs with external computational processing.

Limitations. Our study has several limitations. Our study group consisted primarily of masters and doctoral students at a large engineering university; other populations, such as professional programmers or university faculty, may exhibit different patterns of performance. Likewise, although the assessment spans a variety of algorithm and task types, it cannot capture the full diversity of algorithmic reasoning encountered in practice.

LLM performance is also sensitive to prompt formulation and meta-parameters. To ensure fair comparison between models, we fixed the parameters and did not attempt to optimize performance through prompting techniques. The performance reflects baseline behavior rather than the optimized upper bound of performance.

Finally, our measure of understanding is behavioral; it is defined in terms of observable task performance rather than internal cognitive states or subjective feelings of understanding.

Future Directions. This work suggests several avenues for future research. One avenue is the explicit theoretical modeling of error probability. In our theoretical framework, we elect for simplicity to ignore errors unrelated to understanding, and we assume that an entity who understands at a given level is always capable of the associated tasks. A mathematical model of error in understanding tasks could enable precise modeling of reliability in human-AI collaboration.

More broadly, the proposed hierarchy could inform AI-assisted education and decision support. Adapting the style of AI tutors to the learner and the algorithmic task could aid learning efficiency and interpretability. Furthermore, understanding how algorithmic understanding aligns (or fails to align) between humans and LLMs is increasingly important for large-scale human-AI systems, where improvements are derived from complementary skill sets.

Lastly, this work falls into a broader conversation on understanding in LLMs. We proposed an explicitly task-based and empirically testable definition of understanding, and found that LLMs exhibited high performance along several axes. More generally, we argue that assertions for or against understanding in LLMs are most useful when based on testable, reproducible criteria. We view the development and comparison of such definitions applied to key domains as an important direction for future research.

Conclusion. In this work, we have proposed a task-based formalization of algorithm understanding. Our hierarchy classifies tasks according to seven cognitive processes with varying levels of abstraction. We tested this hierarchy using five algorithms with varying structure, input, and complexity, and evaluated both students and LLMs on tasks related to these algorithms. Our findings suggest that humans and LLMs have competing strengths and weaknesses in algorithmic tasks. Recognizing and modeling these differences is an essential step to building human-AI systems that are reliable, intelligent, and cognitively aligned.

Acknowledgements. The authors would like to thank Santosh Vempala for feedback and guidance in the design of the theoretical framework. With thanks also to Sashank Varma and Anna Ivanova for helpful discussions. This work was funded in part by NSF Award CCF-2106444 and a Simons Investigator award.

References

Aher, G. V., Arriaga, R. I., & Kalai, A. T. (2023). Using large language models to simulate multiple humans and replicate human subject studies. *International conference on machine learning*, 337–371.

Baumberger, C., Beisbart, C., & Brun, G. (2016). What is understanding? an overview of recent debates in epistemology and philosophy of science. *Explaining understanding*, 1–34.

Beattie, V., Collins, B., & McInnes, B. (1997). Deep and surface learning: A simple or simplistic dichotomy? *Accounting education*, *6*(1), 1–12.

Bellemare-Pepin, A., Lespinasse, F., Thölke, P., Harel, Y., Mathewson, K., Olson, J. A., Bengio, Y., & Jerbi, K. (2024). Divergent creativity in humans and large language models. *arXiv preprint arXiv:2405.13012*.

Bloom, B., Engelhart, M., Furst, W., E abd Hill, & Krathwohl, D. (1956). *Taxonomy of educational objectives: The classification of educational goals*. New York: David McKay Company.

Didolkar, A., Goyal, A., Ke, N. R., Guo, S., Valko, M., Lillicrap, T., Rezende, D., Bengio, Y., Mozer, M., & Arora, S. (2024). Metacognitive capabilities of llms: An exploration in mathematical problem solving. *arXiv preprint arXiv:2405.12205*.

Friedman, M. (1974). Explanation and scientific understanding. *the Journal of Philosophy*, *71*(1), 5–19.

Hagendorff, T., Dasgupta, I., Binz, M., Chan, S. C., Lampinen, A., Wang, J. X., Akata, Z., & Schulz, E. (2023). Machine psychology. *arXiv preprint arXiv:2303.13988*.

Koo, R., Lee, M., Raheja, V., Park, J. I., Kim, Z. M., & Kang, D. (2023). Benchmarking cognitive biases in large language models as evaluators. *arXiv preprint arXiv:2309.17012*.

Li, H., Chong, Y. Q., Stepputtis, S., Campbell, J., Hughes, D., Lewis, M., & Sycara, K. (2023). Theory of mind for multi-agent collaboration via large language models. *arXiv preprint arXiv:2310.10701*.

Macmillan-Scott, O., & Musolesi, M. (2024). (ir) rationality and cognitive biases in large language models. *Royal Society Open Science*, *11*(6), 240255.

Mahowald, K., Ivanova, A. A., Blank, I. A., Kanwisher, N., Tenenbaum, J. B., & Fedorenko, E. (2024). Dissociating language and thought in large language models. *Trends in Cognitive Sciences*.

Marton, F., & Säljö, R. (1976). On qualitative differences in learning: I—outcome and process. *British journal of educational psychology*, *46*(1), 4–11.

Mayer, R. E. (2002). Rote versus meaningful learning. *Theory into practice*, *41*(4), 226–232.

Mitchell, M., & Krakauer, D. C. (2023). The debate over understanding in ai's large language models. *Proceedings of the National Academy of Sciences*, *120*(13), e2215907120.

Perrenet, J., Groote, J. F., & Kaasenbrood, E. (2005). Exploring students' understanding of the concept of algorithm: Levels of abstraction. *ACM SIGCSE Bulletin*, *37*(3), 64–68.

Reid, M., & Vempala, S. S. (2025). Does gpt really get it? a hierarchical scale to quantify human and ai's understanding of algorithms. *Proceedings of the AAAI Conference on Artificial Intelligence*, *39*, 1492–1500.

Wilkenfeld, D. A. (2013). Understanding as representation manipulability. *Synthese*, *190*, 997–1016.

Zhao, Y., Zhang, R., Li, W., & Li, L. (2025). Assessing and understanding creativity in large language models. *Machine Intelligence Research*, *22*(3), 417–436.