Benchmarking Generative Models on Computational Thinking Tests in Elementary Visual Programming

Victor-Alexandru Pădurean, Adish Singla

Max Planck Institute for Software Systems, Germany

Motivation and Overview

- Generative models excel in advanced programming benchmarks but struggle with elementary visual programming tasks for school students.
- We introduce a benchmark grounded in visual programming to evaluate computational thinking and problem-solving skills in generative models.
- We developed a large-scale synthetic data generation pipeline to create tasks and explanations for model fine-tuning.
- Our fine-tuned model, LLAMACT, achieves state-of-the-art performance, matching GPT-4o, but still lags behind elementary school students.

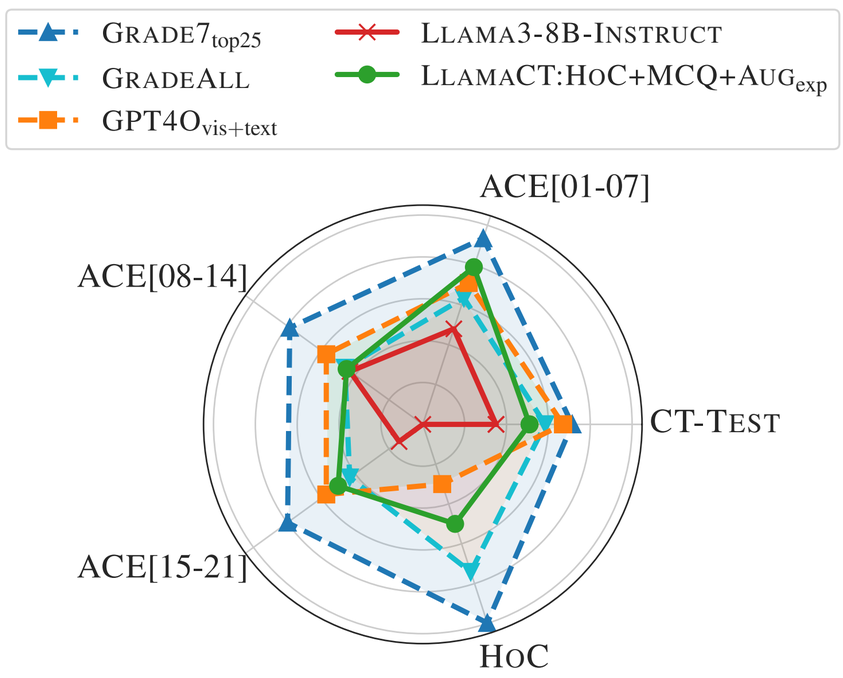
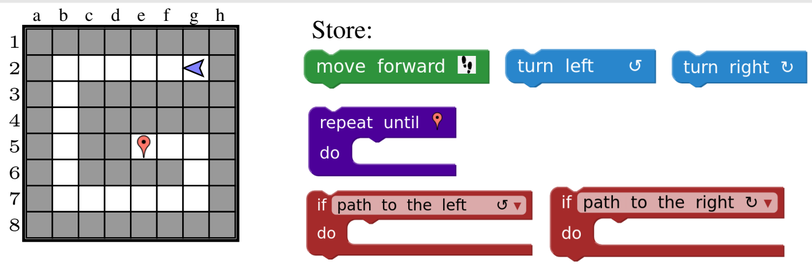


Figure 1: Performance of various models compared to school students.

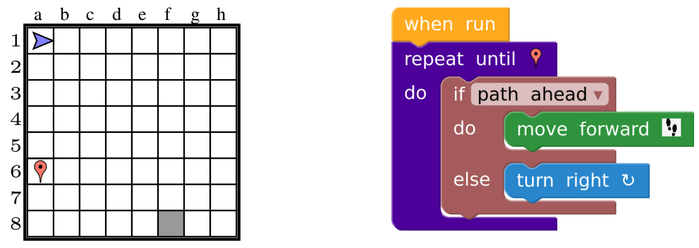
Computational Thinking Tests

- Our benchmark is designed to evaluate diverse computational thinking skills and includes three representative tests: HOC, ACE, and CT-TEST.

HOC-16. Write a solution code in the Workspace that navigates the avatar to the goal. You can only use blocks from the Store for writing code.



ACE-21. You are given a code and an incomplete grid. You can add additional wall cells to the grid by converting any of the free cells into wall cells. What is the smallest number of additional wall cells you must add such that the grid is solved by the code?



OPTION A	1
OPTION B	2
OPTION C	7
OPTION D	8

Figure 2: Examples of tasks from the benchmark.

Our Synthetic Data Generation Pipeline

- Synthetic data includes tasks for solution synthesis, multi-choice questions, and fine-grained skills (basics, tracing, grid synthesis).
- We use symbolic information from code execution to obtain explanations for each answer, enhancing the fine-tuning process.
- We generate over $100,000$ tasks to fine-tune LLAMA3-8B-INSTRUCT, resulting in LLAMACT.

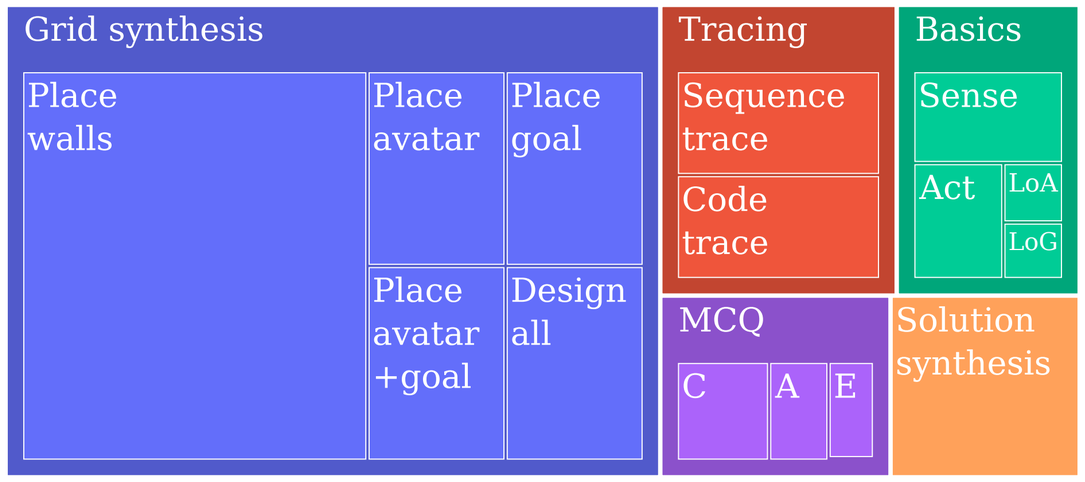


Figure 3: Task distribution across categories in synthetic dataset.

Experimental Results

Technique	HOC	ACE	CT-TEST	Overall
RANDOM	0.0	25.0	25.0	16.7
CODELLAMA-7B-INSTRUCT	0.0 (0.0)	14.3 (0.0)	29.2 (0.0)	14.3 (0.0)
LLAVA1.5-7B	0.0 (0.0)	28.6 (0.0)	20.8 (0.0)	16.7 (0.0)
LLAMA3-8B-INSTRUCT	0.0 (0.0)	34.9 (2.0)	34.7 (5.0)	22.9 (1.3)
GPT4Ovis	20.0 (0.0)	38.1 (3.0)	52.8 (3.0)	36.9 (1.6)
GPT4Otext	30.0 (0.0)	61.9 (3.0)	59.7 (3.0)	50.7 (1.7)
GPT4Ovis+text	30.0 (0.0)	61.9 (0.0)	66.7 (0.0)	53.0 (0.0)
LLAMACT:HOC+MCQ	11.7 (4.0)	44.4 (5.0)	33.3 (5.0)	29.8 (1.2)
LLAMACT:HOC+MCQexp	40.0 (9.0)	43.5 (3.0)	36.1 (2.0)	40.0 (3.6)
LLAMACT:HOC+MCQ+AUGexp	50.0 (4.0)	57.8 (1.0)	51.4 (3.0)	53.0 (1.7)
GRADEALL	74.1	50.9	58.5	61.2
GRADE7top25	99.8	84.0	71.4	85.1

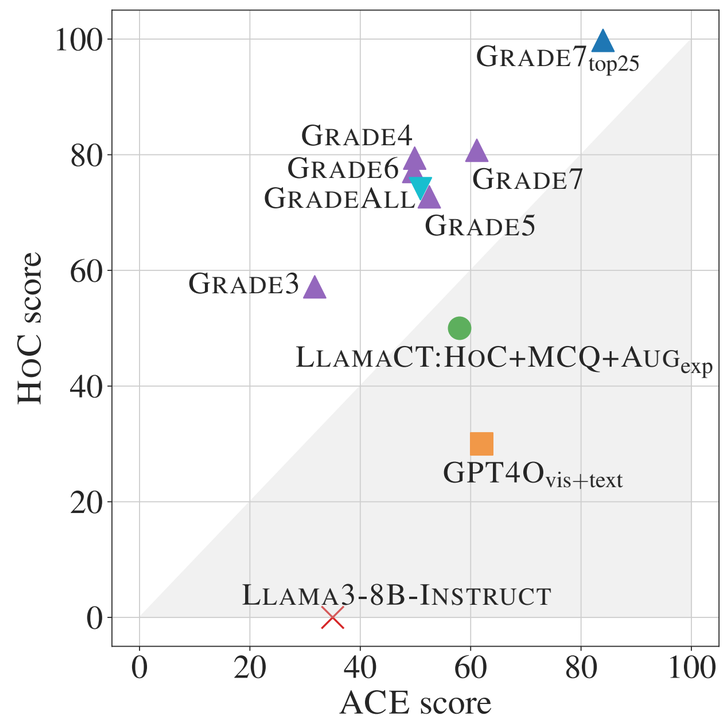


Figure 4: Performance of models on benchmark and comparison with students.

Conclusions

- Substantial performance gains by leveraging fine-grained tasks along with symbolic explanations.
- Models still face challenges in combining spatial, logical, and programming skills to solve the tests.
- Further work required to close the gap with school students.

MAX PLANCK INSTITUTE FOR SOFTWARE SYSTEMS

NEURAL INFORMATION PROCESSING SYSTEMS