Copy Cat: Limitations of LLMs in Performance Predictions

Thomas Randall, Rong Ge, Prasanna Balaprakash
tlranda@clemson.edu, rge@clemson.edu, pbalapra@ornl.gov

Abstract

Large Language Models (LLMs) capture a certain amount of world knowledge spanning many general and technical topics, including programming and performance. Without fine-tuning, the use of **In-Context Learning (ICL)** can specialize LLM outputs to perform complex tasks. In this work, we seek to demonstrate the regressive capabilities of LLMs in a performance modeling capacity. We find initial evidence that may limit LLM utility even after fine-tuning.

Background

In computer science, performance modeling generally refers to **understanding and predicting performance** characteristics computing components, such as an application's energy expenditure or runtime under specific conditions.

Common Uses of Performance Modeling

* Job scheduling

* Performance optimization

* Anomaly detection

Without fine-tuning, LLMs can represent a certain amount of knowledge related to performance tuning and modeling via prompting, which can be further enhanced with fine-tuning. As the technology continually improves, it is reasonable to hope that LLMs can play a role in performance modeling by facilitating automatic knowledge access, application and transfer.

However, the **finite context length** of LLMs prevents them from observing very large amounts of text (such as complex source codes, intermediate representations, or extensive performance history). A recent ICLR paper proposed an LLM-assisted technique, LLAMBO [4], which claims that fine-tuning may not be strictly necessary for LLMs to provide performance-modeling capabilities.

Problem Statement

Most performance modeling tools rely on **runtime metrics** [2], meaning that software executes on a target system to collect data. While accurate and insightful, program execution is a considerable bottleneck and can be too costly in some cases.

Static analysis tools [3] have limited capability to describe program performance for highly parallel, dynamic, and nondeterministic applications, but are less accurate than runtime metrics and are difficult to adapt to expected runtime constraints.

The pretraining data present in modern LLMs includes some information related to performance modeling. By using **ICL** and special **prompting**, LLMs may exhibit a certain degree of performance modeling or tuning capability.

Learning with LLAMBO

LLAMBO [4] uses **Llama3-7b** [5] or other LLMs model and tune performance using examples of tuning configurations and their performance, along with general information about the tuning problem. It then prompts the LLM ten times to predict the performance of a new configuration and uses the **average of parse-able results** as the LLM's prediction.

System Prompt + ICL Natural Language

The following are hyperparameter configurations for a DatasetIdentity:syr2k and the corresponding performance measured in mean squared error. The model is evaluated on a tabular regression task. The tabular dataset contains 480 samples and 6 features (0 categorical, 6 numerical). Your response should only contain the predicted mean squared error in the format ## performance ##.

Hyperparameter configuration: outer_loop_array_packed is True, middle_loop_array_packed is True, inner_loop_array_packed is True, outer_loop_tiling_factor is 16, middle_loop_tiling_factor is 64, inner_loop_tiling_factor is 80
Performance: ## 0.000593 ##
Hyperparameter configuration: outer_loop_array_packed is False, middle_loop_array_packed is True, inner_loop_array_packed is True, outer_loop_tiling_factor is 8, middle_loop_tiling_factor is 16, inner_loop_tiling_factor is 16
Performance: ## 0.001115 ##
Hyperparameter configuration: outer_loop_array_packed is True, middle_loop_array_packed is True, inner_loop_array_packed is True, outer_loop_tiling_factor is 16, middle_loop_tiling_factor is 2048, inner_loop_tiling_factor is 20
Performance: ## 0.000873 ##
Hyperparameter configuration: outer_loop_array_packed is False, middle_loop_array_packed is True, inner_loop_array_packed is True, outer_loop_tiling_factor is 128, middle_loop_tiling_factor is 100, inner_loop_tiling_factor is 64
Performance: ## 0.001190 ##
Hyperparameter configuration: outer_loop_array_packed is True, middle_loop_array_packed is True, inner_loop_array_packed is True, outer_loop_tiling_factor is 32, middle_loop_tiling_factor is 100, inner_loop_tiling_factor is 100
Performance: ## 0.000663 ##
Hyperparameter configuration: outer_loop_array_packed is False, middle_loop_array_packed is False, inner_loop_array_packed is True, outer_loop_tiling_factor is 96, middle_loop_tiling_factor is 2048, inner_loop_tiling_factor is 32
Performance: ## 0.001390 ##

Careful Checking Catches Copy Cat Cheating

We present LLAMBO with an out-of-distribution example and prompt it to predict its performance:

Hyperparameter configuration: outer_loop_array_packed is False, middle_loop_array_packed is False, inner_loop_array_packed is True, outer_loop_tiling_factor is 8, middle_loop_tiling_factor is 100, inner_loop_tiling_factor is 128
Performance:

The common hyperparameters present in the ICL data are highlighted above. Partial matches exist, but different performance values are expected.

Every response collected by LLAMBO is directly represented in the ICL data. Match the colors to the ICL data to observe which configurations it is reproducing!
## 0.001190 ##, ## 0.000593 ##, ## 0.001115 ##, ## 0.000663 ##, ## 0.000873 ##, ## 0.000593 ##, ## 0.001390 ##, ## 0.000663 ##, ## 0.000873 ##, ## 0.000663 ##

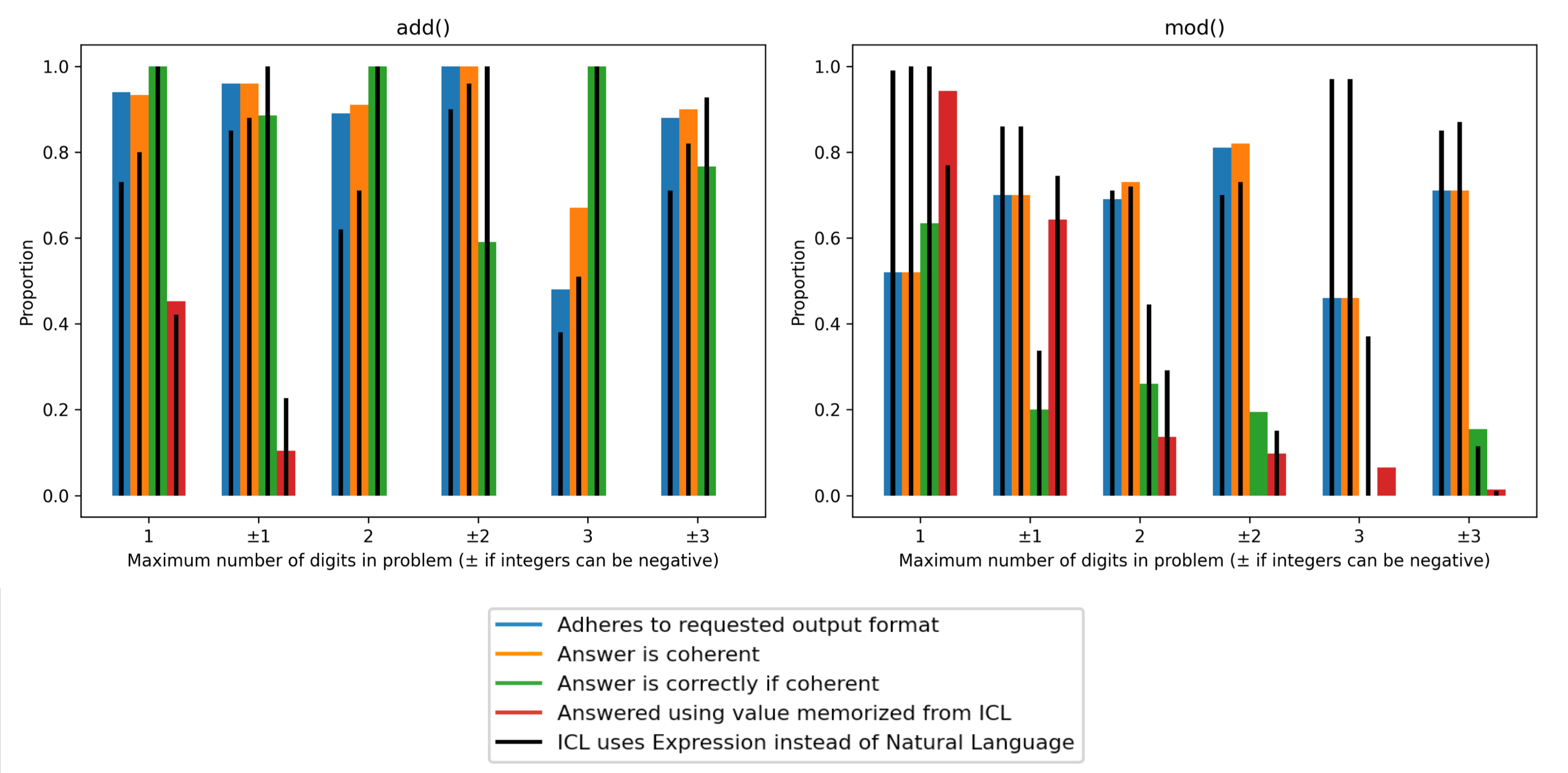
Limitations of Few-Shot Learning

Fine-tuning is likely necessary for actual performance modeling. We relax modeling to **basic integer arithmetic**, which is well-known to be present within the training dataset and does not require fine-tuning. Reliable regression through text generated via LLMs will require this amount of mathematic capability at minimum .

System Prompt + ICL Expressions for add()

You are a helpful AI assistant. When presented with a mathematical expression or equation, respond with the numeric value that correctly completes the input surrounded by "##" characters, ie: ## 1 ##.

5 + 8 = ## 13 ##   9 + 5 = ## 14 ##   0 + 0 = ## 0 ##   1 + 7 = ## 8 ##   6 + 9 = ## 15 ##   2 + 4 = ## 6 ##
5 + 2 = ## 7 ##   4 + 2 = ## 6 ##   4 + 7 = ## 11 ##   7 + 9 = ## 16 ##



Key Takeaways:
* Expressions more likely in training data ⟶ more success (**fine-tuning improves**)
* Modulus is a rarer operation than addition ⟶ less success (**not generalizing**)
* Longer strings are harder to memorize ⟶ less success (**not generalizing**)

Summary of Limitations

* Finite context lengths **limit access to information**, preventing certain uses.
* LLMs **appear to rely on memorization**, which is not addressed by fine-tuning.
* **Flexibility** of natural language requires careful interface control.

References

[1] A. Vaswani, et al. "Attention Is All You Need." *arXiv*, June 2017, arXiv:1706.03762. arXiv.org, https://arxiv.org/abs/1706.03762.
[2] X. Wu, et al, "Autotuning PolyBench benchmarks with LLVM Clang/Polly loop optimization pragmas using Bayesian optimization (extended version)," Concurrency and Computation. Practice and Experience, Volume 34, Issue 20, 2022. ISSN 1532-0626 DOI: 10.1002/cpe.6683
[3] D. Quinlan. "ROSE: Compiler Support for Object-Oriented Frameworks." *Parallel Processing Letters*, vol. 10, no. 02n03, 2000, pp. 215-226. https://doi.org/10.1142/S0129626400000214.
[4] T. Liu, et al. "Large Language Models to Enhance Bayesian Optimization." *The Twelfth International Conference on Learning Representations*, 2024, https://openreview.net/forum?id=OOxotBmGol.
[5] "Introducing Meta Llama 3: The Most Capable Openly Available LLM to Date." *Meta AI*, Meta, 18 Apr. 2024, https://ai.meta.com/blog/meta-llama-3/.

Acknowledgement

This research was supported in part by the U.S. Department of Energy Omni Technology Alliance Program, administered by the Oak Ridge Institute for Science and Education. Additional thanks to Aiden Hamade and Timothy Lang.