E-PAL: Extension of Program-Aided Language Models

Claudia Lee, Jessica Yan, William Zhang
COS484 Final Project

Introduction

Program-aided Language models (PAL) uses LLMs to decompose problems into runnable Python code and offloads the solution to a Python interpreter which then executes the program to produce the final answer. By decoupling reasoning (done by the LLM) from execution (done by the interpreter), PAL eliminates many of the arithmetic and logical slips that plague chain-of-thought solutions. We extend upon this work by testing PAL's ability to solve two new tasks from the BIG-Bench evaluation suite: tracking shuffled objects and web of lies.

Approach

Our study focuses on evaluating Program-aided Language Models on two tasks within the Big Bench hard dataset which are the tracking shuffled objects and web of lies. Both of which require logical reasoning compared to pattern matching. For each task, we designed prompt templates that demonstrate to the large language model how to translate the query into the Python snippets. We then took these python snippets and executed them to get the final answer. Essentially, the model had to be able to handle the decomposition of the problem and we would evaluate how well it did on the code it produced. We then compare this to a baseline where we prompt GPT-4 the same inputs, but do not provide any other instructions other than to provide a final answer for evaluation. The experiments that were conducted were on GPT-4 with a temperature of 0 with no changes in other parameters. We report the accuracy of of GPT-4 with PAL and without PAL and the confusion matrices provide some insight to where PAL falls short.

Visual Diagram of PAL Approach?

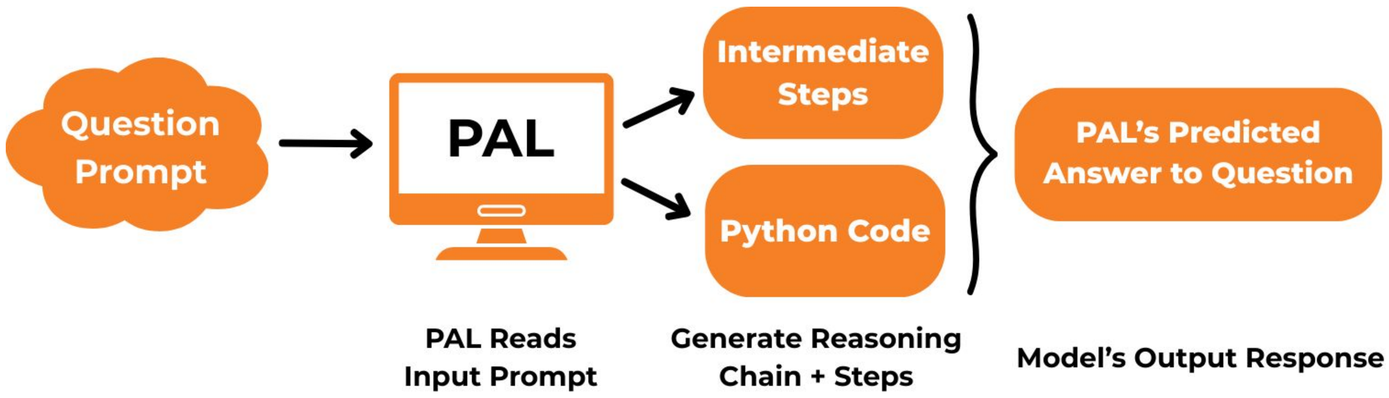

Figure 1: General Approach

Results

Prompt	PAL Accuracy	Without PAL
Tracking Shuffled Objects	0.608	0.519
Web of Lies	1.0	0.596

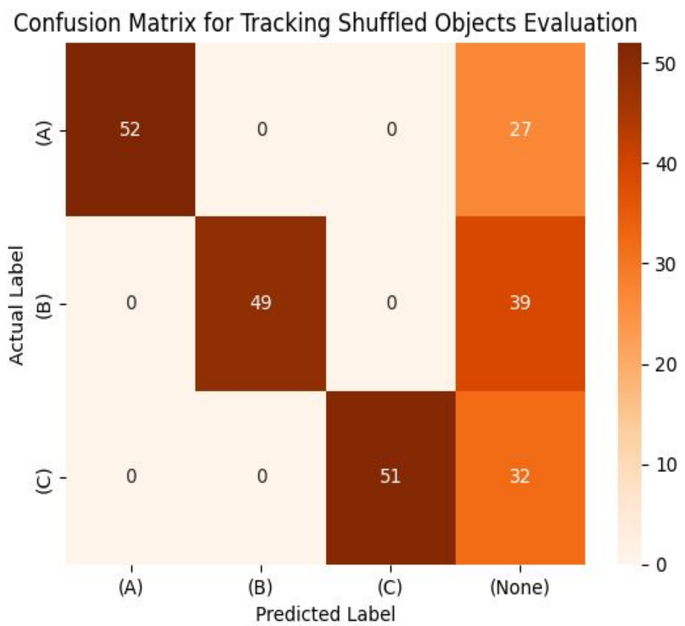

Figure 2: Confusion Matrix for Tracking Shuffled Objects Prompt

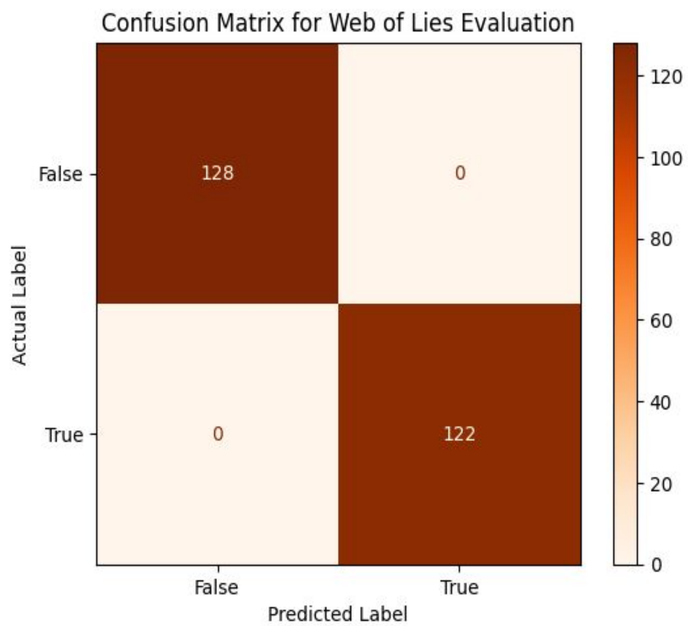

Figure 3: Confusion Matrix for Web of Lies Prompt

Conclusions

Program-aided Language (PAL) consistently reduced the logical errors within the results. Specifically for tracking shuffled objects, it was able to boost the accuracy from 51.9% to 50.8% and provide 100% accuracy on web of lies compared the baseline of 59.6%. Furthermore, in the tracking shuffled objects confusion matrix, we can see that the mistakes commonly within the None section which means there may have been errors in terms of how the Python code was created by GPT-4. For web of lies, there was no issues as PAL was able to fully resolve all the inputs with no errors. These results demonstrate the improvement of large language models through offloading the execution of the final answer to something like Python interpreter. The approach presented provides a way for new datasets to be added and for other similar logical problems that can be solved by GPT-4 with the usage of PAL.

Future Directions

With the current setup and extension we have made through this project, there are various possible next steps. Specifically, one could be to examine how a language model responds to an action of calling itself through a different prompt. This approach would allow us to better understand how effective it may be for a model to call itself when responding to a question. Another approach would be to test it on newer reasoning models that have come about like DeepSeek R1 and evaluate its performance. Finally, another extension would be to identify subsets of prompts that PAL may underperform on and iteratively improve the model in those areas through self-feedback..

Acknowledgments

We would like to thank the following people for their guidance throughout this project:

- Adithya Bhaskar as our adviser
- Rest of COS484 Staff for allowing a rewarding research exploration