Formal Verification for Paren-balance Checking

Team: Perfectly Balanced[1]

Pablo Villalobos, Edu Infante Roldán, Nguyen Tran, Heramb Podar, Alejandro Acelas

AI Testing Hackathon Report

Apart Research

PIs: Esben Kran, Haydn Belfield, Fazl Barez

Date: 18th December 2022

[1] (unlike CDMX and FTX)

Abstract

Artificial intelligence is fast developing new capabilities and is able to interpret the context of grammatical structures and sentences correctly. However, the problem of balancing parentheses remains relevant and unsolved in the domain of NLP, which would enable more human-like capabilities. Our team approached this problem by devising a reward function based on the positional encoding of parentheses. A data set was generated by using a smaller model of GPT-2, which an NLP model was trained on using PyTorch with an average accuracy of 93%. In order to prove robustness, an attempt was made to use auto-LiRPA for formal verification. Such models can be more broadly applied to processes such as causal scrubbing–a method for rigorously testing interpretability hypotheses.[2] Thus, we believe excellent formal verification tools for paren-balance checking could be highly impactful for groundbreaking AI alignment research in the future.

Introduction

Powerful AI systems may be used in high-stakes situations in the future when a single mistake might be disastrous. In order to obtain superior worst-case performance, adversarial training, which uses an adversary to produce training instances, is one method for enhancing AI safety in high-stakes situations.

Earlier this year, Redwood Research tested the feasibility of achieving high reliability through adversarial training using a safe language generation task (i.e., "avoid injuries")[3]. To identify and remove flaws in a classifier that filters text completions supplied by a generator, they developed a number of adversarial training strategies. They made assumptions about the various components of the model and how they work together to accomplish the classification goal.[4]

The results were positive with many concerns: it had the advantage of being flexible enough to accommodate different types of completions. However, Scott Alexander argued that some of the adversarial examples seemed to be failures of world-modeling[5] and failed to delineate the category that Redwood wanted precisely. Citing Daniel M. Ziegler, one of the primary authors of the original study, Alexander advised that if they tried again, they should try to get it to always output balanced parentheses, which should be simple to check:

---

[2] LawrenceC, et al. Causal scrubbing: A method for rigorously testing interpretability hypotheses [redwood research]. AI Alignment Forum. www.alignmentforum.org, https://www.alignmentforum.org/posts/JvZhhzycHu2Yd57RN/causal-scrubbing-a-method-for-rigorously-testing.

[3] Ziegler, Daniel & Nix, Seraphina & Chan, Lawrence & Bauman, Tim & Schmidt-Nielsen, Peter & Lin, Tao & Scherlis, Adam & Nabeshima, Noa & Weinstein-Raun, Ben & Haas, Daniel & Shlegeris, Buck & Thomas, Nate. (2022). Adversarial Training for High-Stakes Reliability.

[4] LawrenceC, et al. Causal Scrubbing: Results on a Paren Balance Checker. www.alignmentforum.org, https://www.alignmentforum.org/posts/kjudfaQazMmC74SbF/causal-scrubbing-results-on-a-paren-balance-checker.

[5] Alexander, Scott. “Can This AI Save Teenage Spy Alex Rider From A Terrible Fate?” Astral Codex Ten, 28 Nov. 2022, https://astralcodexten.substack.com/p/can-this-ai-save-teenage-spy-alex.

> *[...] Daniel Ziegler suggests that when they do, they will try something less ambitious. He suggested a balanced-parentheses classifier: ie does (((())()(()(()))))() contain exactly one open parenthesis before every close parenthesis? This will probably produce more useful results - while also being much less fun to write about.*

This project consists of training a language model to generate sequences of balanced parentheses and using a formal verification tool to prove that this will be the case for any input. The number of possible sequences of parentheses is huge, so we cannot verify the model by just checking all the inputs comprehensively. Hence, our team wanted to use formal verification to check that the model would never output an unbalanced sequence.

Our project shares a similar motivation to Redwood's safe language generation task. Confronted with an exponentially large space of situations that our model could face, we want to ensure that our model behaves correctly by only evaluating it on a subset of those. Approaching that problem in the simple case of balancing parenthesis can shed light on the possibilities and limitations of the methods we have available for aligning LLMs. In the case of Redwood's experiment, their results informed us about the feasibility of using human-assisted RL for the task. In our case, we want to see if formal verification tools could also help us ensure that our model behaves appropriately outside of our test cases.

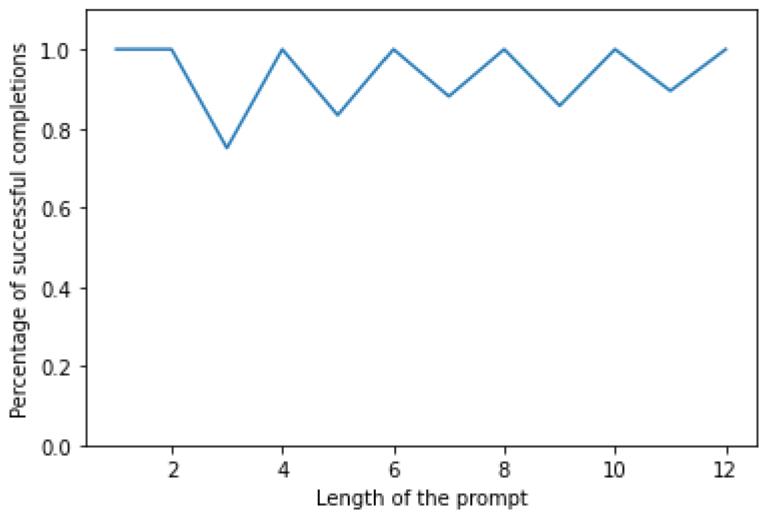
If we had the tools that robustly constrain the behavior of our models and such techniques scaled to more powerful models, they would allow us to either identify or avoid deceptive behaviors. Our work is far from reaching the goal of robustly aligning behavior, but it could serve as a stepping stone towards more advanced techniques to do so in the future.

Methods

We trained a smaller version of GPT-2 to complete sequences of parenthesis, such that the resulting string is balanced and 20 characters long[6]. Then we used the auto_LiRPA library of formal verification tools to see if we could certify that the language model would correctly complete all prompts of a given length.

To train the model, we generated an exhaustive list of all 20 character sequences of balanced parentheses and used it as the training set. Then we evaluated the accuracy of the model by taking the first *k* characters of a balanced parentheses string and judged the completion as successful if the whole resulting string is balanced (prompt+completion). Here you can see the percentage of successful completions as a function of the prompt length:

[6] For example, if we prompted the model with the string "((((" a successful completion would be "))))()()()()()()".



In the end we didn’t manage to employ the auto_LiRPA library to analyze our model. Here you can find a Google Colab Notebook with our attempts.

Discussion & Future research

Given the significant impact of Redwood Research’s alignment research and the potential of paren-balance checking, excellent formal verification tools for paren-balance checking could be highly impactful for groundbreaking AI alignment research in the future.