Algorithm Distillation Interpretability

Summary

Google released a paper called algorithm distillation https://arxiv.org/abs/2210.14215 which claims that their setup allows the model to learn to do reinforcement learning in context. The idea of this project is to reproduce the AD setup (or something similar enough to be interesting) and do mechanistic interpretability research to try to figure out whether the models actually learn to do RL in context or something simpler, and if so how does that actually work and what that can tell us about inner optimization and agency on models.

Introduction

A lot of discussions and disagreements in alignment are ultimately generated by intuitions about things like agency and optimization in ML systems, training dynamics, mesa optimizers, etc.
I think we are deeply confused about this topic and that researching toy models might lead us to get some insights, or at least more clear evidence for some view or another.
This is especially important if it turns out something similar to big transformers ends up leading to AGI but might also just give us general insights that transfer to future models or lead us to develop skills and techniques that will help research future models.

DeepMind's algorithm distillation paper claims their setup lets a transformer model learn to do reinforcement learning in context by training it on sequences of multiple episodes from multiple tasks. Other people like Sam Marks are more sceptical.
I think that it would be great if we can actually look at the model and try to figure out a mechanistic explanation of what it is actually doing and why.
Is Sam Marks right that the sort of exploration that the AD agent does is "when you don't know what to do, mimic an incompetent RL agent"? Does that change when we modify things like model size and amount of tasks?
This project is not trivial but I think its reasonable especially the models involved are relatively small since the paper uses 4 layer transformers and we can probably try making even smaller ones, and if that doesn't work I'll take it as some evidence that there's something interesting to learn about the model.

Plan summary and experiments

Now, how are we going to actually do this? I think we'll mostly figure out along the way and I don't want to make overly complicated plans before actually getting feedback from reality to see which direction is interesting, but I think this broad outline is reasonable:

- Train a transformer using the algorithm distillation setup on a series of toy task maybe using minigrid or some other set of environments and try to get similar results than the original AD paper, and experiment with models of different sizes and configurations.

- Adapt Neel Nanda's TransformerLens library to work in our setup (I expect this will probably just involve modifying the embeddings).
- Create some interactive way of running experiments and generating synthetic inputs for the model with something like Joseph Bloom's interactive App for interpreting decision transformers. In fact, we may decide to use Joseph Bloom's code as a starting point since AD is pretty similar to decision transformers (as you can think of an AD model as a specific kind of multi episode DT model).
- Look into the existing literature for techniques and experiments to run and try to come up with new ones.
- Run whichever experiments we come up with and go on iteratively from there.

The running experiments part is pretty vague because the experiments we choose to do will be informed by the results we get and there is no standard way of doing mechanistic interpretability research yet. There are some existing techniques like Causal scrubbing, but I think MI research is mostly about running a lot of experiments with tight feedback loops, focusing on whatever experiments and techniques seem useful for understanding the model.

Results

In the best case scenario of this project, we would gain a mechanistic understanding of how algorithm distillation works in a way that leads to interesting insights into things like agents and optimization in transformers. It might also be that you can get good results with this task without doing anything interesting, but I think that would also be an interesting insight in itself. It could also be the case that it does in-context RL but not in a way that generalises to bigger models and environments.
At minimum, I expect this to be a good opportunity for people to try mechanistic interpretability research, acquire the necessary skills, and test their fit with research in the field.

Output

Blogpost or paper about our findings.

Risks and downsides

Like a lot of interpretability research this might have the negative side effect of advancing capabilities.
I think it is unlikely we actually align AI without making a lot of progress in knowing what we are doing and understanding the probably scrutable but currently not understood giant matrices, and the field of ML is likely going to put a lot of money and effort into finding the same insights via trial and error and grad student descent anyway so I think MI is mostly net positive.
Doing things with better OPSEC like Conjecture seems to be aiming for would probably be better, and if we were planning to make big SOTA models or something I would be more worried, but for projects like this is probably okay if we don't worry excessively about that.

Another plausible risk is that, since as far as I can tell there are no open source implementations of AD, publishing our code might accelerate that line of research.
But I think the solution to this is just not make it open source until there's other replications out there first.

I think those are reasonable concerns and that we should likely discuss whether we should publish our results if we for example found a better than SOTA RL algorithm inside the model or something along those lines (that specific example seems pretty unlikely to me though).

Team

Team size
Will mostly depend on how many people are interested in working on the project and in helping with organization.
3-5 sounds like a reasonable number, but take that as an estimate.

Time commitment
I'm willing to work full-time on this, especially if we get funding (more on that later), but I'm open to different levels of commitment from group members, although preferably you should be able to dedicate 10h a week.
Aiming for the project to last 3-4 months to mostly mirror the AISC structure, but I'm open to changing that if people have better ideas.

Research Lead
Víctor Levoso Fernández. Contact me at victorlevosofernandez@gmail.com, or in the new MI Discord.
I did a master's degree in computer science before deciding to focus on AI alignment. I'm currently trying to do Mechanistic interpretability research because that seems the most promising.
I participated in the SERI MATS online training program in the mechanistic interpretability stream, and got started working on this and similar projects like interpretability in decision transformers.

Skill requirements

I think that as long as you have basic CS skills and basic ML or math knowledge you can probably help.
Mechanistic interpretability is pretty new and I don't expect anyone to be already an expert on it, though being knowledgeable about ML and or having Pytorch skills definitely helps.
Neel's guide to getting started with MI is probably a good summary of the skills required to work on this kind of problem:
https://www.neelnanda.io/mechanistic-interpretability/getting-started
But if you have doubts about whether you can help, I encourage you to indicate your interest on the Discord anyway, and there are probably easier projects you can help with instead if you really don't have the skills necessary.

I think this project will be a mix of engineering work and more conceptual coming up with experiments to try and hypothesize about what is going on inside models.
I’m hoping to find team members who can help both with the actual implementation and with ideas for what interpretability techniques to try, but it's fine if you can only do one of them. The grunt of the work is probably going to be writing ML code, but I think it would be fine if we have some people with more math and conceptual research skills, even if they aren't great at programming.