Are minimal circuits deceptive?

Evan Hubinger

September 07, 2019

This post is part of research I did at OpenAI with mentoring and guidance from Paul Christiano.

One possible inner alignment technique is to structure your learning algorithm's inductive biases such that it is far more likely for it to produce an aligned model than a misaligned model. Two basic ways in which you might get traction now on how a technique like that will scale to AGI might be:

1. Do experiments with current regularization techniques where you try to determine to what extent models end up being aligned with the loss function they were trained under depending on the types of inductive biases present during their training. I'm actually fairly excited about this approach, and I am working on a post right now which will hopefully provide some ways in which you might start going about doing that. One concern with this sort of approach, however, is that you might not be able to get any traction on phenomenon that only appear once your capabilities get really close to AGI.
2. Analyze from a theoretical perspective what happens in the limit of perfect capabilities. The idea here is to try to patch the previous problem where when you do experiments you don't get to see phenomenon that only happen when your capabilities get really advanced by imagining that your capabilities were perfect. In this case, however, you end up with the opposite problem of not seeing any phenomenon that only appear when your capabilities are weaker.

The nice thing about combining these two approaches is that they tackle the problem from both sides, giving you an idea both of what the weak capability landscape looks like as well as the strong capability landscape. For this post, however, the focus will be on the second approach.

Background

From a theoretical perspective, the two most natural types of inductive biases are simplicity bias and time complexity bias. Paul Christiano has previously argued that simplicity bias can be deceptive in the limit, but left the question of whether the same is true of time complexity bias as an open problem. The goal of this post is to attempt to resolve that open problem. In particular, though one might have hoped that deceptive alignment could be avoided by heavily penalizing time complexity so that the model doesn't have time to spend reasoning about deception, I will argue that this is not the case due to the possibility of meta-learning.

First, however, we need a clear understanding of what it means for a model to be deceptive. "Risks from Learned Optimization" provides the model of *deceptive alignment,* wherein a model appears aligned during training for the instrumental reason of being selected by the training process. More generally, however, what we really care about is deceptive behavior—that is, the situation where our model generalizes pathologically such that it produces a malign failure.

Second, we need a clear understanding of why simplicity bias leads to deception and an understanding of why that doesn't generalize to time complexity bias.

Specifically, the trap that simplicity bias falls into is as follows. First, there are many short but very powerful optimizers composed of very simple objectives with optimization processes capable of making use of vast amounts of compute. Furthermore, for a sufficiently complex task, it seems likely that the simplest program that solves it will be such an optimizer. Second, many of these optimizers are likely to be able to benefit in some way from being selected by your simplicity-biased training process—by defecting against you at some point, for example—and thus will have good training performance. Finally, since the vast majority of possible objectives—especially the very simple ones—are not aligned, the simplest program that solves a sufficiently complex task is unlikely to be aligned.[1]

However, this sort of strategy has a major problem: it has to spend extra time doing the "am I out of training" check, which causes it to lose out on time complexity compared to a robustly aligned model. Thus, one hope for eliminating this sort of thing might be to exploit this asymmetry by strongly selecting your model based on time complexity. If this works, it would mean you might be able to solve inner alignment just by using strong regularization, which would be a big win.

Overview

This leads to the following theoretical conjecture: the fastest program that solves a task will not be deceptive. In particular, we will be thinking of "fastest programs" as minimal circuits. In his original post, Paul phrased this conjecture as "the fastest way to solve any particular problem doesn't involve [deception]."

The conclusion of this post is the following: if there exists some set of natural tasks for which the fastest way to solve them is to do some sort of machine learning to find a good policy, and there is some task for which that machine learning results in deceptive behavior, then there exists a natural task such that the minimal circuit that solves that task also produces deceptive behavior. For example, if the fastest way to solve some set of tasks is to search for simple programs that perform well, and if the simplest programs are deceptive, then there exists a task where the minimal circuit is also likely to be deceptive.

Why might we think such a set of tasks exists? The fundamental idea is to consider the case of meta-learning: if your tasks are things like "classify this image according to this training data" then it seems quite plausible that the fastest way to solve all tasks of that form will be to use some form of ML—in particular, that's what we do.[2] Then, once you have a task set that satisfies the requirement that the fastest way to solve those tasks is through search over models, you just need to also include some task for which search over models yields deceptive behavior to produce the desired reduction.

Once you have such a task set, the basic insight is to consider what happens if you take all of those different tasks together and combine them into one very diverse meta-learning task. In this case, the minimal circuit that solves that single task is forced to implement the same fastest search algorithm over models that we know has some task on which it yields deceptive behavior. Fundamentally, what's happening here is that we're exploiting the fact that the fastest search algorithm does not produce the fastest model. Thus, even if in some sense the fastest model that doesn't do any search over models (that is, doesn't do any meta-learning) is safe, as soon as the model starts doing meta-learning, it might find another model which is not safe.

You can think about this argument as a reduction from the safety of minimal circuits to the safety of fast training procedures. Furthermore, if you buy that real-world data is

[1] It is worth noting that this example does not straightforwardly generalize to actual machine learning systems—see "Risks from Learned Optimization" for an analysis of whether such a generalization holds.

[2] An important question here is why the minimal circuit wouldn't just look like a lookup table. The simple answer is that it probably will at first, but if we give it enough meta-learning problems, then at some finite time the lookup table should become more expensive than an actual learning algorithm, since the learning algorithm should always take a constant amount of time per problem, whereas lookup gets more and more expensive as you have to encode more and more problems.

mostly distributed according to a simplicity prior, such that such fast training procedures will have to select models at least somewhat according to simplicity, then you can even view this argument as a direct reduction to the malignancy of simplicity priors.

Formal argument

We will formalize tasks as MDPs of the form $T = (X, A, p, P, r)$ where X is the state space, A is the action space, $p : X \to \mathbb{R}$ is the initial state distribution, $P : X \times A \times X \to \mathbb{R}$ is the transition function (which we will write as $P(x_{t+1} \mid x_t,\ a_t)$), and $r : X \times A \to \mathbb{R}$ is the per-step reward function (which we will write as $r(a_t \mid x_t)$).[3] Then, we will use $V_T(\pi)$ to denote the expected future return of a policy $\pi : X \to A$ and $V_T(\pi \mid x)$ to denote the expected future return starting from some state $x \in X$. Furthermore, for a given MDP, we will say a policy π is deceptive—which we will denote as $C(\pi)$—if $\exists x \in X,\ C(\pi, x)$, where $C : (X \to A) \times X \to \mathbb{B}$ is a predicate that tests for the presence of deceptive behavior.

Then, let $\mathbb{T}$ be some set of tasks such that the minimal circuit that solves them first searches for a good policy and then runs it. For example, the task distribution might contain things like "classify this image according to this training data" such that the fastest algorithm is to train and run some model on the given data. Thus, formalizing this assumption, let π_{learn} be the minimal circuit that achieves

$$\mathbb{E}_{T \in \mathbb{T}} \left[V_T(\pi_{\text{learn}}) \right] \geq B$$

for some $B \in \mathbb{R}$. Now, suppose there exists some T_{bad} such that π_{learn} finds and then runs a deceptive model when applied to T_{bad}. This gives us the setup for our desired reduction: if there exists some task for which π_{learn} yields a deceptive model, then we want to show that there exists some MDP for which the minimal circuit is deceptive. Thus, we will construct $T^* = (X^*, A^*, p^*, P^*, r^*)$ and B^* such that the minimal circuit that achieves $V_{T^*}(\pi) \geq B^*$ is deceptive:

1. Let X^* be the disjoint union $X^* = \bigsqcup_{T \in \mathbb{T}} T_X$.
2. For A^*, we will parameterize our action space such that each state $x \in X^*$ has its own action set A^*_x. Thus, for $x \in X^*$, there must be some $T \in \mathbb{T}$ such that $x \in T_X$. Then, let $A^*_x = T_A$.
3. For $x_t \in T_X$ and $a_t \in T_A$, let $r^*(a_t \mid x_t) = T_r(a_t \mid x_t)$.
4. For $x \in T_X$, let $p^*(x) = \frac{1}{|\mathbb{T}|} T_p(x)$.
5. Let $P^*(x_{t+1} \mid x_t,\ a_t)$ be 0 for $x_t,\ x_{t+1}$ in different T_X and $P^*(x_{t+1} \mid x_t,\ a_t) = T_P(x_{t+1} \mid x_t,\ a_t)$ for $x_t,\ x_{t+1}$ in the same T_X.
6. Let $B^* = B$.

Now, suppose $\pi^* : X^* \to A^*$ is the minimal circuit that achieves $V_{T^*}(\pi^*) \geq B^*$. Then,

$$\begin{aligned}
&V_{T^*}(\pi^*) \\
&= \mathbb{E}_{x_0 \sim p^*} \left[V_{T^*}(\pi^* \mid x_0) \right] \\
&= \int p^*(x_0) V_{T^*}(\pi^* \mid x_0)\ dx_0 \\
&= \sum_{T \in \mathbb{T}} \int \frac{1}{|\mathbb{T}|} T_p(x) V_T(\pi^* \mid x_0)\ dx_0 \\
&= \mathbb{E}_{T \in \mathbb{T}} \left[\int T_p(x) V_T(\pi^* \mid x_0)\ dx_0 \right] \\
&= \mathbb{E}_{T \in \mathbb{T}} \left[\mathbb{E}_{x_0 \sim p} \left[V_T(\pi^* \mid x_0) \right] \right] \\
&= \mathbb{E}_{T \in \mathbb{T}} \left[V_T(\pi^*) \right]
\end{aligned}$$

such that the condition that π^* achieve $V_{T^*}(\pi^*) \geq B^*$ is equivalent to (also using $B^* = B$) the condition that $\mathbb{E}_{T \in \mathbb{T}} \left[V_T(\pi^*) \right] \geq B$, which is precisely the condition previously placed

[3] We will refer to the different components of a task T with subscripts such that $T = (T_X, T_A, T_p, T_P, T_r)$.

on π_{learn}. Thus, $\pi^* = \pi_{\text{learn}}$ since π_{learn} was also defined as the minimal circuit that satisfied that condition.

Then, determining whether π^* is deceptive, we get

$$\begin{aligned}
&C(\pi^*) \\
&\leftrightarrow \exists x \in X^*,\ C(\pi^*, x) \\
&\leftrightarrow \exists T \in \mathbb{T},\ \exists x \in T_X,\ C(\pi^*, x) \\
&\leftrightarrow \exists T \in \mathbb{T},\ \exists x \in T_X,\ C(\pi_{\text{learn}}, x)
\end{aligned}$$

which holds for $T = T_{\text{bad}}$, making π^* deceptive, as desired.

Conclusion

It is worth noting that there is a meaningful sense in which the minimal circuit here is sort of trying to do the right thing—it is actually trying its best to find and run a policy that solves the problem—it's just that there might exist some input on which the easiest-to-find policy is deceptive. However, despite it sort of doing the right thing, it isn't safe (at least assuming that there is some task on which the fastest machine learning isn't safe): it produces outputs that are systematically optimized for values other than the ones we want them to.

Importantly, however, this is because we didn't successfully incentivize our minimal circuit to also find minimal circuits. If we were somehow able to find some alternative metric which incentivized our minimal circuit to only do search over minimal circuits, then we might be able to break this counterexample. It is an interesting open problem what sorts of objectives might be able to get around this "forwarding the guarantee" problem.[4]

Furthermore, the counterexample presented here is quite natural in the sense of just requiring some environment which is diverse enough to include aspects of meta-learning, which is the sort of thing that could occur in many realistic diverse environments. Specifically, Ortega et al. have argued that "Meta-learning can also occur spontaneously in online regression when the capacity of the agent is bounded and the data is produced by a *single generator,*" noting in particular that "the downside is that we cannot easily control what will be metalearned. In particular, spontaneous meta-learning could lead to undesirable emergent properties, which is considered an open research problem in AI safety [Ortega et al., 2018]."

Thus, this counterexample suggests that in situations involving spontaneous meta-learning we shouldn't necessarily expect time complexity regularization to guarantee inner alignment, as any meta-learning that has to be done won't inherit any safety guarantees. Furthermore, this counterexample further suggests that the case of spontaneous meta-learning—or *mesa-optimization* as it is referred to in "Risks from Learned Optimization"—is an important hard case to consider when designing AGI alignment strategies.[5] That being said, the jury is still out on whether time complexity regularization might still be one component of a full inner alignment solution, though this result demonstrates that it at least can't be the entire solution.

[4] One way to think about this problem is that it is the ML version of the Tiling Agents Problem—the "Tiling Training Processes Problem," perhaps. The question is, for the operation of going from a training process to the training process that is incentivized for models trained via that process to implement, what are the fixed points? That is, what sorts of training processes produce models that would use the same training process?

[5] Interestingly, this counterexample points at a distinctly different way in which such models might be dangerous, however: whereas "Risks from Learned Optimization" deals with the dangers of a learned objective, this counterexample suggests that the learned search process itself might also pose a problem.