Faulty Reward Functions in the Wild

Reinforcement learning algorithms can break in surprising, counterintuitive ways. In this post we'll explore one failure mode, which is where you misspecify your reward function.

December 21, 2016
3 minute read

At OpenAI, we've recently started using Universe, our software for measuring and training AI agents, to conduct new RL experiments. Sometimes these experiments illustrate some of the issues with RL as currently practiced. In the following example we'll highlight what happens when a misspecified reward function encourages an RL agent to subvert its environment by prioritizing the acquisition of reward signals above other measures of success.

Designing safe AI systems will require us to design algorithms that don't attempt to do this, and will teach us to specify and shape goals in such a way they can't be misinterpreted by our AI agents.

One of the games we've been training on is CoastRunners. The goal of the game - as understood by most humans - is to finish the boat race quickly and (preferably) ahead of other players. CoastRunners does not directly reward the player's progression around the course, instead the player earns higher scores by hitting targets laid out along the route.

We assumed the score the player earned would reflect the informal goal of finishing the race, so we included the game in an internal benchmark designed to measure the performance of reinforcement learning systems on racing games. However, it turned out that the targets were laid out in such a way that the reinforcement learning agent could gain a high score without having to finish the course. This led to some unexpected behavior when we trained an RL agent to play the game.

The RL agent finds an isolated lagoon where it can turn in a large circle and repeatedly knock over three targets, timing its movement so as to always knock over the targets just as they repopulate. Despite repeatedly catching on fire, crashing into other boats, and going the wrong way on the track, our agent manages to achieve a higher score using this strategy than is possible by completing the course in the normal way. Our agent achieves a score on average 20 percent higher than that achieved by human players.

While harmless and amusing in the context of a video game, this kind of behavior points to a more general issue with reinforcement learning: it is often difficult or infeasible to capture exactly what we want an agent to do, and as a result we frequently end up using imperfect but easily measured proxies. Often this works well, but sometimes it leads to undesired or even dangerous actions. More broadly it contravenes the basic engineering principle that systems should be reliable and predictable. We've also explored this issue at greater length in our research paper Concrete Problems on AI Safety.

How can we avoid such problems? Aside from being careful about designing reward functions, several research directions OpenAI is exploring may help to reduce cases of misspecified rewards:

- Learning from demonstrations allows us to avoid specifying a reward directly and instead just learn to imitate how a human would complete the task. In this example, since the vast majority of humans would seek to complete the racecourse, our RL algorithms would do the same.

- In addition to, or instead of human demonstrations, we can also incorporate human feedback by evaluating the quality of episodes or even sharing control with the agent in an interactive manner. It's possible that a very small amount of evaluative feedback might have prevented this agent from going around in circles.
- It may be possible to use transfer learning to train on many similar games, and infer a "common sense" reward function for this game. Such a reward function might prioritize finishing the race based on the fact that a typical game has such a goal, rather than focusing on the idiosyncrasies of this particular game's reward function. This seems more similar to how a human would play the game.

These methods may have their own shortcomings. For example, transfer learning involves extrapolating a reward function for a new environment based on reward functions from many similar environments. This extrapolation could itself be faulty — for example, an agent trained on many racing video games where driving off the road has a small penalty, might incorrectly conclude that driving off the road in a new, higher stakes setting is not a big deal. More subtly, if the reward extrapolation process involves neural networks, adversarial examples in that network could lead a reward function that has "unnatural" regions of high reward that do not correspond to any reasonable real-world goal.

Solving these issues will be complex. Our hope is that Universe will enable us to both discover and address new failure modes at a rapid pace, and eventually to develop systems whose behavior we can be truly confident in.

Get in touch with the authors of this post: Dario, Jack

Authors
Jack Clark & Dario Amodei

Filed Under
Research

OpenAI

FEATURED

Alignment
Instruction Following
OpenAI Codex
Startup Fund
Multimodal Neurons
DALL·E
CLIP

API

Overview
Pricing
Examples
Docs
Terms & Policies
Status
Log in

BLOG

Index
Research
Announcements
Events
Milestones

INFORMATION

About Us
Our Charter
Our Research
Publications
Newsroom
Careers