Handling adversarial errors

Paul Christiano Follow
Apr 15, 2015 · 10 min read

Even a very powerful learning system can't do everything perfectly at first—it requires time to learn. Any proposal for AI control needs cope with mistakes while learning. In this post I'll describe why this problem is non-trivial, and in the next post I'll offer some more constructive answers.

Regret

It's convenient to imagine a "training" period during which a machine learning system learns, and a "test" period during which it is guaranteed to perform well. But unless (and even if) you make a heroic effort, there will be features of the domain that just don't appear in your training data. And so no matter how good your learner is, and no matter how thorough your training is, you can never realistically say that "now the model is trained—it's all clear sailing from here."

That said, in many contexts we *can* prove guarantees like "the learner won't make too many mistakes in total," for a sufficiently careful definition of "mistake." That is: the learner may take a long time to learn the whole domain, if some aspects of it just don't appear in the data for a long time. But in that case, the learner's ignorance can't do any damage.

The possibility of errors at test time presents a challenge for AI control. It's hard to say much about when these errors occur, and so if our systems make use of supervised learning, they may fail at inopportune times. If we can't cope with adversarial errors, then:

- In order to prove that our systems will work, we need to make some additional assumptions about their behavior. But we don't have any good candidates, and I think there are fundamental obstacles to finding any.

- There may actually be serious problems that we haven't anticipated. For example, failure to cope with "adversarial" errors can be in indication that your training data may not be rich enough. What's more, at the end of this post, I'll describe two reasons that these errors could actually occur at the worst possible time.

I see three possible responses:

1. Either find an alternative to the supervised learning paradigm.
2. Develop a much deeper understanding of supervised learning paradigm.
3. Make systems that are robust to a small number of adversarial failures.

I think that [1] and [2] are problematic. The supervised learning paradigm seems to be extremely effective in practice (note that this problem also afflicts conventional approaches to semi-supervised learning), so giving up on it seems to be a massive handicap. Moreover, we don't seem to be anywhere close to the kind of understanding that would let us control the errors that learners make while learning, and I'm not sure if it's even possible to do so (that is, I'm not sure if the statements we'd want to prove are even true).

If possible, I think that [3] would be by far the least demanding and most useful option.

What is "one" error?

If a system makes only "one" error, that *doesn't* mean that its output differs from the "right" output on only one occasion.

From the learner's perspective, an output is only an "error" if it leads to suboptimal payoff. So all we can say is that the learner won't receive a suboptimal payoff too often before it gets the picture. This could involve any number of "bad" decisions, if those bad decisions don't lead to bad payoffs.

For example, suppose that we train a system for a month, and then release a static version of it into the world. We can make no guarantees at all about the behavior of the resulting system in the real world (though we might be able to make such guarantees if we can justify strong assumptions about the training data). If there is some

difference between the training data and the test data, this learner may just keep messing things up, over and over again. We may not have trained the function we expect to have trained; the training data may fail to pin down some key ambiguity that we didn't consider.

In order to prove a rigorous bound on the number of mistakes, we need to actually provide the learner with feedback telling it when it made a mistake, and the learner needs to continue to learn from that feedback.

Counterfactual oversight

If we use counterfactual oversight, and the learner receives feedback on 1% of their decisions chosen at random, then "one" mistake corresponds to 99 bad decisions (in expectation) followed by one last bad decision that finally receives negative feedback. An upper bound of "ten" mistakes corresponds to 1000 bad decisions, and so on.

This shouldn't be too surprising—if we give the learner training data very rarely, it takes it a long time to learn.

This consideration seems to be the biggest reason to adopt a significant probability of oversight. But thinking through this issue suggests more effective approaches than purely stochastic oversight. I'll discuss this in a future post.

An illustration

Consider my recent proposal for safely applying supervised learning. This proposal is completely compromised by adversarial failures, and hence a completely unsatisfactory proposal.

For a simple case, suppose that I ask such an AI "Is the number of primes between 1 and 8000 even or odd?"

To evaluate its answer, I ask about the the number of primes between 1 and 4000, and between 4001 and 8000. I check the answers for consistency, and give the original answer a good reward iff the answers are consistent.

This process then continues recursively: at each step we choose one of the two questions from the preceding step, and then we compute its rating in the same way. Finally, in the last step the learner makes a claim of the form "X is prime," and we check this claim by hand, assigning it a high score iff it is true.

If our learner never makes any errors, then it's easy to verify by induction that every question must be answered correctly.

But suppose that our learner is only guaranteed to make at most 1 error. Then our final result can be completely wrong. In fact, the situation is even more dire.

For example, suppose that the learner is wrong about whether 1 is prime. Then the learner is guaranteed to make at most one mistake, but our answer is guaranteed to be wrong. Moreover, with probability 7999/8000, the learner won't realize the error of their ways. And if we do the same thing again, the learner will be wrong again. Even if the learner is split 50–50 about whether 1 is prime, they will make on average 8000 errors before they figure if out. But it seems like they should only have to make 1 error. That's a big gap (and it would be easy for it to be even bigger).

Interpretation

I think this isn't just a formal artifact—the fact that our system breaks down under adversarial errors is a real sign that it can't work. There may be other algorithms that work in practice but fail under adversarial errors. But I think that "failure under adversarial errors" is generally a sign that we should either try to find some assumptions that rule out the particular failure that causes trouble, or else be skeptical of the system.

What's going on in the context of my proposal?

We are generating training data by walking down a random tree. But all of the "action" may happen in a very small part of that tree, and so the learner may never get feedback about the really important issues.

To overcome this problem, we need some system for directing the learner's attention to the data points that are most likely to be surprising.

Proving a system works, even under "adversarial" failures, guarantees that we've addressed this issue. It also guarantees that we've addressed any similar issues that we might not have thought of yet.

Malignant failure modes

(Warning: the rest of this post is very weird.)

Vulnerability to adversarial errors is a troubling sign which *suggests* that a proposal may have flaws that we just haven't identified.

But in some contexts, errors may actually *be* adversarial, in which case vulnerability to adversarial errors is itself a serious problem. I'll illustrate two cases. Both of these seem somewhat outlandish. I don't know if this concern is fundamentally limited to outlandish scenarios, and I don't know if anything this outlandish could actually happen.

Evolution

Suppose that human reproduction worked differently: humans are surrounded by challenges (like big game to hunt) and every time someone solves one, they get to have one offspring. In this environment, it seems like humans may well have evolved the same kinds of complex psychology and culture that they have in the real world, with a love of solving challenges that is similar-in-kind to humans' love of sex or delicious food.

Imagine that this environment was engineered by a race of aliens, who are breeding humans to be useful for solving similar challenges that appear in the real world. But it turns out that humans care about all sorts of things other than solving these challenges. So if these humans learn that they are being used, and learn that there is some real-world significance to the challenges they are solving, it wouldn't be too surprising if they decided to choose a suboptimal answer on a challenge in order to have a desired effect on the outside world (for example, they might answer a challenge in a way that caused them to escape from their artificial environment and gain access to the outside world).

This may seem somewhat outlandish, but we can imagine similar results from very sophisticated versions of contemporary supervised learning (or even unsophisticated versions, scaled up to a massive enough amount of hardware that they could produce human-level intelligence).

Many supervised learning systems work by considering a large class of possible models, and then using the models from that class that have worked well in the past.

Usually this is not done using an explicit evolutionary dynamic, but in any case it shares many of the essential features of evolution: we explore a number of models, and then "keep" the one or ones that seem to perform well in some training environment.

One way a model can "win" in this game is by being the kind of model that has the desired properties—that will continue to do well in any environment similar to the training environment. But another way that it can win is by having a "survival instinct," and recognizing that performance in the training environment is the best way to survive.

Such a model will continue to perform well as long as its survival is contingent on doing well in the training environment. But when selection pressure is removed, or when there are other paths to survival, it may suddenly start behaving differently. If so, its behavior may be strategically optimized for its own long-term survival, rather than for continuing to perform well in the testing environment.

Simulations

Bayesian reasoning can also be used to solve many supervised learning problems. The hope is that the "real" hypothesis is assigned some reasonable probability by the Bayesian's prior. Every time they make a prediction worse than the "real" hypothesis, the probability of the real hypothesis grows. And so the number total number of "mistakes" is bounded.

But there can be "skeptical" hypotheses, which are very different from the real one yet make identical predictions in almost all cases. Because almost no evidence bear on these hypotheses, if they start out almost as likely as the real hypothesis, then they will remain almost as likely indefinitely.

One particularly plausible skeptical hypothesis is the simulation hypothesis: that what we perceive as reality is a computer program being run a computer. There is a reasonable philosophical argument in favor of the simulation hypothesis, and a small but significant fraction of people think that it is plausible or probable.

Under the simulation hypothesis, the simulated universe looks just like a simple lawful universe—until the simulation stops, or the simulators decide to change its character. This could lead to very strange behavior by rational agents who expect such a change to happen soon.

Fixing the problem

If we apply either of the above AI's to solve a problem we care about, we may get surprising results at an inopportune time. But if our application is robust to adversarial errors, even these pessimistic scenarios can't cause trouble.

Suppose that an AI decides to behave strangely, because it expects that the simulation is going to be shut down in a few minutes anyway. When the simulations isn't shut down, the AI will learn a lesson, and update against the simulation hypothesis. If the AI started out by assigning a very high probability to the simulation hypothesis, then this could happen a few times. But as long as "I'm not in a simulation" is accorded some reasonable prior probability (as it will be by any non-dogmatic prior), this can only happen so many times before the AI will get the picture.

Similarly, suppose that we've run an exhaustive search to find an AI that seems to perform well, but is actually going to behave very strangely when the stakes get high. The AI can do this, but by doing so it decreases its own fitness. As long as our selection process was also *capable* of identifying an AI that actually just plays the game it was trained to play, and as long as we keep applying selection power "in the wild," then eventually the intended behavior will win out. Every time that another contender decides to pull a punch, the honest AI will pull a bit further ahead. (Of course, this can fail in any number of ways—for example, by getting stuck in a local optimum.)

In any case, a wacky situation can lead to a bit of wacky behavior, but there is a bound on how much of it there can be. So if our schemes are resistant to a bit of wacky behavior, then we can ride it out and wait for things to settle down to normality.

We have bigger problems if our AI is very confident that it is in a simulation, or if our evolution procedure was unable to find any agent that will reliably perform well (rather than behaving strangely after a context change). But these extreme scenarios correspond to actual **failures of the learning algorithms**, reflected in violations of their purprted guarantees. So we can hope to rule this out by design better learning algorithms. Designing good learning algorithms may end up being the hardest part of the AI control problem, but for now I'm happy to set it aside—I would be satisfied if we could reduce AI control to the problem of building learning systems that satisfy intuitive formal guarantees.