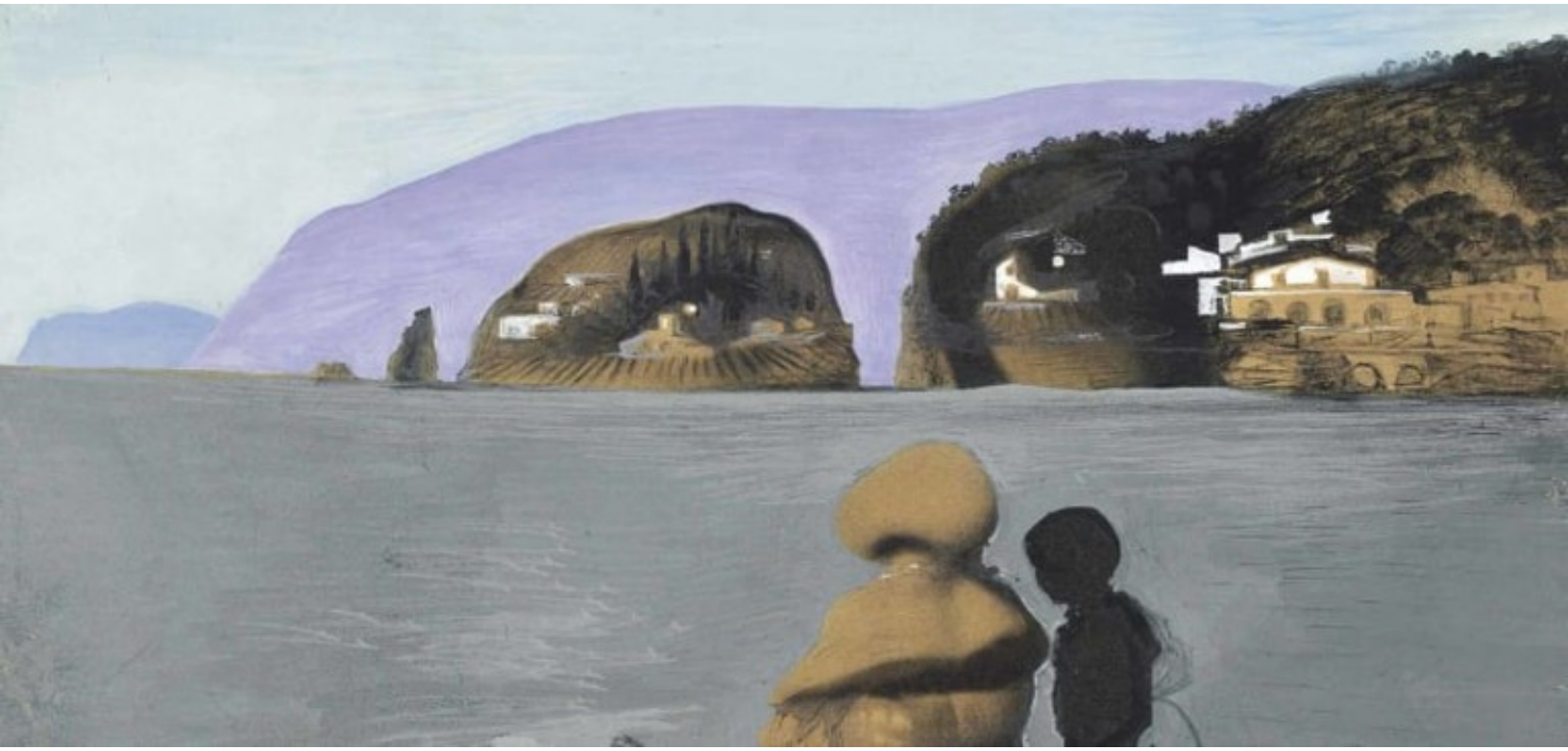

Partial Agency

1. Partial Agency
2. The Parable of Predict-O-Matic
3. Random Thoughts on Predict-O-Matic
4. Defining Myopia
5. The Credit Assignment Problem
6. Bayesian Evolving-to-Extinction

Partial Agency

Crossposted from the AI Alignment Forum. May contain more technical jargon than usual.

Epistemic status: very rough intuitions here.

I think there's something interesting going on with Evan's notion of myopia.

Evan has been calling this thing "myopia". Scott has been calling it "stop-gradients". In my own mind, I've been calling the phenomenon "directionality". Each of these words gives a different set of intuitions about how the cluster could eventually be formalized.

Stop-Gradients

Nash equilibria are, abstractly, modeling agents via an equation like

$a^* = \text{argmax}_a f(a, a^*)$. In words: a^* is the agent's mixed strategy. The payoff $f(.,.)$ is a function of the mixed strategy in two ways: the first argument is the causal channel, where actions directly have effects; the second argument represents the "acausal" channel, IE, the fact that the other players know the agent's mixed strategy and this influences their actions. The agent is maximizing across the first channel, but "ignoring" the second channel; that is why we have to solve for a fixed point to find Nash equilibria. This motivates the notion of "stop gradient": if we think in terms of neural-network type learning, we're sending the gradient through the first argument but not the second. (It's a kind of mathematically weird thing to do!)

Myopia

Thinking in terms of iterated games, we can also justify the label "myopia". Thinking in terms of "gradients" suggests that we're doing some kind of training involving repeatedly playing the game. But we're training an agent to play as if it's a single-shot game: the gradient is rewarding behavior which gets more reward within the single round even if it compromises long-run reward. This is a weird thing to do: why implement a training regime to produce strategies like that, if we believe the nash-equilibrium model, IE we think the other players will know our mixed strategy and react to it? We can, for example, win chicken by going straight more often than is myopically rational. Generally speaking, we expect to get better rewards in the rounds after training if we optimized for non-myopic strategies during training.

Directionality

To justify my term "directionality" for these phenomena, we have to look at a different example: the idea that "when beliefs and reality don't match, we change our beliefs". IE: when optimizing for truth, we optimize "only in one direction". How is this possible? We can write down a loss function, such as Bayes' loss, to define accuracy of belief. But how can we optimize it only "in one direction"?

We can see that this is the same thing as myopia. When training predictors, we only consider the efficacy of hypotheses one instance at a time. Consider *supervised learning:* we have "questions" $x_1, x_2, \ldots$ etc and are trying to learn "answers" $y_1, y_2, \ldots$ etc. If a neural network were somehow able to mess with the training data, it would not have much pressure to do so. If it could give an answer on instance x_1 which improved its ability to answer on x_2 by manipulating y_2, the gradient would not specially favor this. Suppose it is possible to take some small hit (in log-loss terms) on y_1 for a large gain on y_2. The large gain for x_2 would not reinforce the specific neural patterns responsible for making y_2 easy (only the patterns responsible for successfully taking advantage of the easiness). The small hit on x_1 means there's an incentive not to manipulate y_2.

It is *possible* that the neural network learns to manipulate the data, if by chance the neural patterns which shift x_1 are the same as those which successfully exploit the manipulation at x_2. However, this is a fragile situation: if there are other neural sub-patterns which are equally capable of giving the easy answer on x_2, the reward gets spread around. (Think of these as parasites taking advantage of the manipulative strategy without doing the work necessary to sustain it.) Because of this, the manipulative sub-pattern may not "make rent": the amount of positive gradient it gets may not make up for the hit it takes on x_1. And all the while, neural sub-patterns which do better on x_1 (by refusing to take the hit) will be growing stronger. Eventually they can take over. This is exactly like myopia: strategies which do better in a specific case are favored for that case, despite global loss. The neural network fails to successfully coordinate with itself to globally minimize loss.

To see why this is also like stop-gradients, think about the loss function as $l(w, w^*)$: the neural weights w determine loss through a "legitimate" channel (the prediction quality on a single instance), plus an "illegitimate" channel (the cross-instance influence which allows manipulation of y_2 through the answer given for x_1). We're optimizing through the first channel, but not the second.

The difference between supervised learning and reinforcement learning is just: reinforcement learning explicitly tracks helpfulness of strategies across time, rather than assuming a high score at x_2 has to do with only behaviors at x_2! As a result, RL can coordinate with itself across time, whereas supervised learning cannot.

Keep in mind that this is a good thing: the algorithm may be "leaving money on the table" in terms of prediction accuracy, but this is exactly what we want. We're trying to make the map match the territory, not the other way around.

Important side-note: *this argument obviously has some relation to the question of how we should think about inner optimizers and how likely we should expect them to be. However, I think it is not a direct argument against inner optimizers. (1) The*

emergence of an inner optimizer is exactly the sort of situation where the gradients end up all feeding through one coherent structure. Other potential neural structures cannot compete with the sub-agent, because it has started to intelligently optimize; few interlopers can take advantage of the benefits of the inner optimizer's strategy, because they don't know enough to do so. So, all gradients point to continuing the improvement of the inner optimizer rather than alternate more-myopic strategies. (2) Being an inner optimizer is non synonymous with non-myopic behavior. An inner optimizer could give myopic responses on the training set while internally having less-myopic values. Or, an inner optimizer could have myopic but very divergent values. Importantly, an inner optimizer need not take advantage of any data-manipulation of the training set like that I've described; it need not even have access to any such opportunities.

The Partial Agency Paradox

I've given a couple of examples. I want to quickly give some more to flesh out the clusters as I see them:

- As I said, myopia is "partial agency" whereas foresight is "full agency". Think of how an agent with high time-preference (ie steep temporal discounting) can be money-pumped by an agent with low time-preference. But the limit of no-temporal-discounting-at-all is not always well-defined.
- An updatefull agent is "partial agency" whereas updatelessness is "full agency": the updateful agent is failing to use some channels of influence to get what it wants, because it already knows those things and can't imagine them going differently. Again, though, full agency seems to be an idealization we can't quite reach: we don't know how to think about updatelessness in the context of logical uncertainty, only more- or less- updatefull strategies.
- I gave the beliefs←territory example. We can also think about the values→territory case: when the world differs from our preferences, we change the world, not our preferences. This has to do with avoiding wireheading.
- Similarly, we can think of examples of corrigibility -- such as respecting an off button, or avoiding manipulating the humans -- as partial agency.
- Causal decision theory is more "partial" and evidential decision theory is less so: EDT wants to recognize more things as legitimate channels of influence, while CDT claims they're not. Keep in mind that the math of causal intervention is closely related to the math which tells us about whether an agent wants to manipulate a certain variable -- so there's a close relationship between CDT-vs-EDT and wireheading/corrigibility.

I think people often take a pro- or anti- partial agency position: if you are trying to one-box in Newcomblike problems, trying to cooperate in prisoner's dilemma, trying to define logical updatelessness, trying for superrationality in arbitrary games, etc... you are generally trying to remove barriers to full agency. On the other hand, if you're trying to avert instrumental incentives, make sure an agent allows you to change its values, or doesn't prevent you from pressing an off button, or doesn't manipulate human values, etc... you're generally trying to add barriers to full agency.

I've historically been more interested in dropping barriers to full agency. I think this is partially because I tend to assume that full agency is what to expect in the long run, IE, "all agents want to be full agents" -- evolutionarily, philosophically, etc. Full agency should result from instrumental convergence. Attempts to engineer partial agency for

specific purposes feel like fighting against this immense pressure toward full agency; I tend to assume they'll fail. As a result, I tend to think about AI alignment research as (1) needing to understand full agency much better, (2) needing to mainly think in terms of aligning full agency, rather than averting risks through partial agency.

However, in contrast to this historical view of mine, I want to make a few observations:

- Partial agency sometimes seems like *exactly what we want,* as in the case of map←territory optimization, rather than a crude hack which artificially limits things.
- Indeed, partial agency of this kind seems *fundamental to full agency.*
- Partial agency seems *ubiquitous in nature*. Why should I treat full agency as the default?

So, let's set aside pro/con positions for a while. What I'm interested in at the moment is ***the descriptive study of partial agency as a phenomenon.*** I think this is an organizing phenomenon behind a lot of stuff I think about.

The partial agency paradox is: why do we see partial agency naturally arising in certain contexts? Why are agents (so often) myopic? Why have a notion of "truth" which is about map←territory fit but not the other way around? Partial agency is a weird thing. I understand what it means to optimize something. I understand how a selection process can arise in the world (evolution, markets, machine learning, etc), which drives things toward maximization of some function. Partial optimization is a comparatively weird thing. Even if we can set up a "partial selection process" which incentivises maximization through only some channels, wouldn't it be blind to the side-channels, and so unable to enforce partiality in the long-term? Can't someone always come along and do better via full agency, no matter how our incentives are set up?

Of course, I've already said enough to suggest a resolution to this puzzle.

My tentative resolution to the paradox is: you don't build "partial optimizers" by taking a full optimizer and trying to add carefully balanced incentives to create indifference about optimizing through a specific channel, or anything like that. (Indifference at the level of the selection process does not lead to indifference at the level of the agents evolved by that selection process.) Rather, *partial agency is what selection processes incentivize by default*. If there's a learning-theoretic setup which incentivizes the development of "full agency" (whatever that even means, really!) I don't know what it is yet.

Why?

Learning is basically episodic. In order to learn, you (sort of) need to do the same thing over and over, and get feedback. Reinforcement learning tends to assume ergodic environments so that, no matter how badly the agent messes up, it eventually re-enters the same state so it can try again -- this is a "soft" episode boundary. Similarly, RL tends to require temporal discounting -- this also creates a soft episode boundary, because things far enough in the future matter so little that they can be thought of as "a different episode".

So, like map←territory learning (that is, epistemic learning), we can kind of expect any type of learning to be myopic to some extent.

This fits the picture where full agency is an idealization which doesn't really make sense on close examination, and partial agency is the more real phenomenon. However, this is absolutely not a conjecture on my part that all learning algorithms produce partial agents of some kind rather than full agents. There may still be frameworks which allow us to approach full agency in the limit, such as taking the limit of diminishing discount factors, or considering asymptotic behavior of agents who are able to make precommitments. We may be able to achieve some aspects of full agency, such as superrationality in games, without others.

Again, though, my interest here is more to understand what's going on. The point is that it's actually really easy to set up incentives for partial agency, and not so easy to set up incentives for full agency. So it makes sense that the world is full of partial agency.

Some questions:

- To what extent is it really true that settings such as supervised learning disincentivize strategic manipulation of the data? Can my argument be formalized?
- If thinking about "optimizing a function" is too coarse-grained (a supervised learner doesn't exactly minimize prediction error, for example), what's the best way to revise our concepts so that partial agency becomes obvious rather than counterintuitive?
- Are there better ways of characterizing the partiality of partial agents? Does myopia cover all cases (so that we can understand things in terms of time-preference), or do we need the more structured stop-gradient formulation in general? Or perhaps a more causal-diagram-ish notion, as my "directionality" intuition suggests? Do the different ways of viewing things have nice relationships to each other?
- Should we view partial agents as multiagent systems? I've characterized it in terms of something resembling game-theoretic equilibrium. The 'partial' optimization of a function arises from the price of anarchy, or as it's known around lesswrong, Moloch. Are partial agents really bags of full agents keeping each other down? This seems a little true, to me, but also doesn't strike me as the most useful way of thinking about partial agents. For one thing, it takes full agents as a necessary concept to build up partial agents, which seems wrong to me.
- What's the relationship between the selection process (learning process, market, ...) and the type of partial agents incentivised by it? If we think in terms of myopia: given a type of myopia, can we design a training procedure which tracks or doesn't track the relevant strategic influences? If we think in terms of stop-gradients: we can take "stop-gradient" literally and stop there, but I suspect there is more to be said about designing training procedures which disincentivize the strategic use of specified paths of influence. If we think in terms of directionality: how do we get from the abstract "change the map to match the territory" to the concrete details of supervised learning?
- What does partial agency say about inner optimizers, if anything?
- What does partial agency say about corrigibility? My hope is that there's a version of corrigibility which is a perfect fit in the same way that map←territory optimization seems like a perfect fit.

Ultimately, the concept of "partial agency" is probably confused. The partial/full clustering is very crude. For example, it doesn't make sense to think of a non-wireheading agent as "partial" because of its refusal to wirehead. And it might be odd to consider a myopic agent as "partial" -- it's just a time-preference, nothing special. However, I do think I'm pointing at a phenomenon here, which I'd like to understand better.

The Parable of Predict-O-Matic

Crossposted from the AI Alignment Forum. May contain more technical jargon than usual.

I've been thinking more about partial agency. I want to expand on some issues brought up in the comments to my previous post, and on other complications which I've been thinking about. But for now, a more informal parable. (Mainly because this is easier to write than my more technical thoughts.)

This relates to oracle AI and to inner optimizers, but my focus is a little different.

1

Suppose you are designing a new invention, a predict-o-matic. It is a wonderous machine which will predict everything for us: weather, politics, the newest advances in quantum physics, you name it. The machine isn't infallible, but it will integrate data across a wide range of domains, automatically keeping itself up-to-date with all areas of science and current events. You fully expect that once your product goes live, it will become a household utility, replacing services like Google. (*Google* only lets you search the *known*!)

Things are going well. You've got investors. You have an office and a staff. These days, it hardly even feels like a start-up any more; progress is going well.

One day, an intern raises a concern.

"If everyone is going to be using Predict-O-Matic, we can't think of it as a passive observer. Its answers will shape events. If it says stocks will rise, they'll rise. If it says stocks will fall, then fall they will. Many people will vote based on its predictions."

"Yes," you say, "but Predict-O-Matic is an impartial observer nonetheless. It will answer people's questions as best it can, and they react however they will."

"But --" the intern objects -- "Predict-O-Matic will see those possible reactions. It knows it could give several different valid predictions, and different predictions result in different futures. It has to decide which one to give *somehow*."

You tap on your desk in thought for a few seconds. "That's true. But we can still keep it objective. It could pick randomly."

"Randomly? But some of these will be huge issues! Companies -- no, nations -- will one day rise or fall based on the word of Predict-O-Matic. When Predict-O-Matic is making a prediction, it is *choosing* a future for us. We can't leave that to a coin flip! We have to select the prediction which results in the best overall future. Forget being an impassive observer! We need to teach Predict-O-Matic human values!"

You think about this. The thought of Predict-O-Matic deliberately steering the future sends a shudder down your spine. But what alternative do you have? The intern isn't suggesting Predict-O-Matic should *lie,* or bend the truth in any way -- it answers 100% honestly to the best of its ability. But (you realize with a sinking feeling) honesty still leaves a lot of wiggle room, and the consequences of wiggles could be huge.

After a long silence, you meet the interns eyes. "Look. People have to trust Predict-O-Matic. And I don't just mean they have to *believe* Predict-O-Matic. They're bringing this thing into their homes. They have to trust that Predict-O-Matic *is something they should be listening to*. We can't build value judgements into this thing! If it ever came out that we had coded a value function into Predict-O-Matic, a value function which selected *the very future itself* by selecting which predictions to make -- we'd be done for! No matter how honest Predict-O-Matic remained, it would be seen as a manipulator. No matter how beneficent its guiding hand, there are always compromises, downsides, questionable calls. No matter how careful we were to set up its values -- to make them moral, to make them humanitarian, to make them politically correct and broadly appealing -- ***who are we to choose?*** No. We'd be done for. They'd hang us. We'd be toast!"

You realize at this point that you've stood up and started shouting. You compose yourself and sit back down.

"But --" the intern continues, a little more meekly -- "You can't just ignore it. The system is faced with these choices. It still has to deal with it somehow."

A look of determination crosses your face. "Predict-O-Matic will be objective. It is a machine of prediction, is it not? Its every cog and wheel is set to that task. So, the answer is simple: it will make whichever answer minimizes projected predictive error. There will be no exact ties; the statistics are always messy enough to see to that. And, if there are, it will choose alphabetically."

"But--"

You see the intern out of your office.

2

You are an intern at PredictCorp. You have just had a disconcerting conversation with your boss, PredictCorp's founder.

You try to focus on your work: building one of Predict-O-Matic's many data-source-slurping modules. (You are trying to scrape information from something called "arxiv" which you've never heard of before.) But, you can't focus.

Whichever answer minimizes prediction error? First you think it isn't so bad. You imagine Predict-O-Matic always forecasting that stock prices will be fairly stable; no big crashes or booms. You imagine its forecasts will favor middle-of-the-road politicians. You even imagine mild weather -- weather forecasts themselves don't influence the weather much, but surely the *collective* effect of *all* Predict-O-Matic decisions will have some influence on weather patterns.

But, you keep thinking. Will middle-of-the-road economics and politics really be the *easiest* to predict? Maybe it's better to strategically remove a wildcard company or two, by giving forecasts which tank their stock prices. Maybe extremist politics are more predictable. Maybe a well-running economy gives people more freedom to take unexpected actions.

You keep thinking of the line from Orwell's *1984* about the boot stamping on the human face forever, except it isn't because of politics, or spite, or some ugly feature

of human nature, it's *because a boot stamping on a face forever is a nice reliable outcome which minimizes prediction error*.

Is that really something Predict-O-Matic would do, though? Maybe you misunderstood. The phrase "minimize prediction error" makes you think of entropy for some reason. Or maybe information? You always get those two confused. Is one supposed to be the negative of the other or something? You shake your head.

Maybe your boss was right. Maybe you don't understand this stuff very well. Maybe when the inventor of Predict-O-Matic and founder of PredictCorp said "it will make whichever answer minimizes projected predictive error" they *weren't* suggesting something which would literally kill all humans just to stop the ruckus.

You might be able to clear all this up by asking one of the engineers.

3

You are an engineer at PredictCorp. You don't have an office. You have a cubicle. This is relevant because it means interns can walk up to you and ask stupid questions about whether entropy is negative information.

Yet, some deep-seated instinct makes you try to be friendly. And it's lunch time anyway, so, you offer to explain it over sandwiches at a nearby cafe.

"So, Predict-O-Matic maximizes predictive accuracy, right?" After a few minutes of review about how logarithms work, the intern started steering the conversation toward details of Predict-O-Matic.

"Sure," you say, "Maximize is a strong word, but it optimizes predictive accuracy. You can actually think about that in terms of log loss, which is related to infor--"

"So I was wondering," the intern cuts you off, "does that work in both directions?"

"How do you mean?"

"Well, you know, you're optimizing for accuracy, right? So that means two things. You can change your prediction to have a better chance of matching the data, or, you can change the data to better match your prediction."

You laugh. "Yeah, well, the Predict-O-Matic isn't really in a position to change data that's sitting on the hard drive."

"Right," says the intern, apparently undeterred, "but what about data that's not on the hard drive yet? You've done some live user tests. Predict-O-Matic collects data on the user while they're interacting. The user might ask Predict-O-Matic what groceries they're likely to use for the following week, to help put together a shopping list. But then, the answer Predict-O-Matic gives will have a big effect on what groceries they really do use."

"So?" You ask. "Predict-O-Matic just tries to be as accurate as possible given that."

"Right, right. But that's the point. The system has a chance to manipulate users to be more predictable."

You drum your fingers on the table. "I think I see the misunderstanding here. It's this word, *optimize*. It isn't some kind of magical thing that makes numbers bigger. And you shouldn't think of it as a person trying to accomplish something. See, when Predict-O-Matic makes an error, an optimization algorithm *makes changes within Predict-O-Matic* to make it learn from that. So over time, Predict-O-Matic makes fewer errors."

The intern puts on a thinking face with scrunched up eyebrows after that, and we finish our sandwiches in silence. Finally, as the two of you get up to go, they say: "I don't think that really answered my question. The learning algorithm is optimizing Predict-O-Matic, OK. But then in the end you get a strategy, right? A strategy for answering questions. And the strategy is trying to do something. I'm not anthropomorphising!" The intern holds up their hands as if to defend physically against your objection. "My question is, this strategy it learns, will it manipulate the user? If it can get higher predictive accuracy that way?"

"Hmm" you say as the two of you walk back to work. You meant to say more than that, but you haven't really thought about things this way before. You promise to think about it more, and get back to work.

4

"It's like how everyone complains that politicians can't see past the next election cycle," you say. You are an economics professor at a local university. Your spouse is an engineer at PredictCorp, and came home talking about a problem at work *that you can understand*, which is always fun.

"The politicians can't have a real plan that stretches beyond an election cycle because the voters are watching their performance *this* cycle. Sacrificing something today for the sake of tomorrow means they underperform today. Underperforming means a competitor can undercut you. So you have to sacrifice all the tomorrows for the sake of today."

"Undercut?" your spouse asks. "Politics isn't economics, dear. Can't you just explain to your voters?"

"It's the same principle, *dear*. Voters pay attention to results. Your competitor points out your under-performance. Some voters will understand, but it's an idealized model; pretend the voters just vote based on metrics."

"Ok, but I still don't see how a 'competitor' can always 'undercut' you. How do the voters know that the other politician would have had better metrics?"

"Alright, think of it like this. You run the government like a corporation, but you have just one share, which you auction off --"

"That's neither like a government nor like a corporation."

"Shut up, this is my new analogy." You smile. "It's called a decision market. You want people to make decisions for you. So you auction off this share. Whoever gets control of the share gets control of the company for one year, and gets dividends based on how well the company did that year. Assume the players are bidding rationally. Each person bids based on what they expect they could make. So the highest bidder is the

person who can run the company the best, and they can't be out-bid. So, you get the best possible person to run your company, and they're incentivized to do *their* best, so that they get the most money at the end of the year. *Except* you can't have any strategies which take longer than a year to show results! If someone had a strategy that took two years, they would have to over-bid in the first year, taking a loss. But then they have to under-bid on the second year if they're going to make a profit, and--
"

"And they get undercut, because someone figures them out."

"Right! Now you're thinking like an economist!"

"Wait, what if two people cooperate across years? Maybe we can get a good strategy going if we split the gains."

"You'll get undercut for the same reason one person would."

"But what if-"

"Undercut!"

After that, things devolve into a pillow fight.

5

"So, Predict-O-Matic doesn't learn to manipulate users, because if it were using a strategy like that, a competing strategy could undercut it."

The intern is talking to the engineer as you walk up to the water cooler. You're the accountant.

"I don't really get it. Why does it get undercut?"

"Well, if you have a two-year plan.."

"I get that example, but Predict-O-Matic doesn't work like that, right? It isn't sequential prediction. You don't see the observation right after the prediction. I can ask Predict-O-Matic about the weather 100 years from now. So things aren't cleanly separated into terms of office where one strategy does something and then gets a reward."

"I don't think that matters," the engineer says. "One question, one answer, one reward. When the system learns whether its answer was accurate, no matter how long it takes, it updates strategies relating to that one answer alone. It's just a delayed payout on the dividends."

"Ok, yeah. Ok." The intern drinks some water. "But. I see why you can undercut strategies which take a loss on one answer to try and get an advantage on another answer. So it won't lie to you to manipulate you."

"I for one welcome our new robot overlords," you but in. They ignore you.

"But what I was really worried about was self-fulfilling prophecies. The prediction manipulates its own answer. So you don't get undercut."

"Will that ever really be a problem? Manipulating things with one shot like that seems pretty unrealistic," the engineer says.

"Ah, self-fulfilling prophecies, good stuff" you say. "There's that famous example where a comedian joked about a toilet paper shortage, and then there really was one, because people took the joke to be about a real toilet paper shortage, so they went and stocked up on all the toilet paper they could find. But if you ask me, money is the real self-fulfilling prophecy. It's only worth something because we think it is! And then there's the government, right? I mean, it only has authority because everyone expects everyone else to give it authority. Or take common decency. Like respecting each other's property. Even without a government, we'd have that, more or less. But if no one expected anyone else to respect it? Well, I bet you I'd steal from my neighbor if everyone else was doing it. I guess you could argue the concept of property breaks down if no one can expect anyone else to respect it, it's a self-fulfilling prophecy just like everything else..."

The engineer looks worried for some reason.

6

You don't usually come to this sort of thing, but the local Predictive Analytics Meetup announced a social at a beer garden, and you thought it might be interesting. You're talking to some PredictCorp employees who showed up.

"Well, how does the learning algorithm actually work?" you ask.

"Um, the actual algorithm is proprietary" says the engineer, "but think of it like gradient descent. You compare the prediction to the observed, and produce an update based on the error."

"Ok," you say. "So you're not doing any exploration, like reinforcement learning? And you don't have anything in the algorithm which tracks what happens *conditional* on making certain predictions?"

"Um, let's see. We don't have any exploration, no. But there'll always be noise in the data, so the learned parameters will jiggle around a little. But I don't get your second question. Of course it expects different rewards for different predictions."

"No, that's not what I mean. I'm asking whether it tracks the probability of *observations* dependent on *predictions*. In other words, if there is an opportunity for the algorithm to manipulate the data, can it notice?"

The engineer thinks about it for a minute. "I'm not sure. Predict-O-Matic keeps an internal model which has probabilities of events. The answer to a question isn't really separate from the expected observation. So 'probability of observation depending on that prediction' would translate to 'probability of an event given that event', which just has to be one."

"Right," you say. "So think of it like this. The learning algorithm isn't a general loss minimizer, like mathematical optimization. And it isn't a consequentialist, like reinforcement learning. It makes predictions," you emphasize the point by lifting one finger, "it sees observations," you lift a second finger, "and it shifts to make future predictions more similar to what it has seen." You lift a third finger. "It doesn't try

different answers and select the ones which tend to get it a better match. You should think of its output more like an *average* of everything it's seen in similar situations. If there are several different answers which have self-fulfilling properties, it will average them together, not pick one. It'll be uncertain."

"But what if historically the system has answered one way more often than the other? Won't that tip the balance?"

"Ah, that's true," you admit. "The system can fall into attractor basins, where answers are somewhat self-fulfilling, and that leads to stronger versions of the same predictions, which are even more self-fulfilling. But there's no guarantee of that. It depends. The same effects can put the system in an orbit, where each prediction leads to different results. Or a strange attractor."

"Right, sure. But that's like saying that there's not always a good opportunity to manipulate data with predictions."

"Sure, sure." You sweep your hand in a gesture of acknowledgement. "But at least it means you don't get purposefully disruptive behavior. The system can fall into attractor basins, but that means it'll more or less reinforce existing equilibria. Stay within the lines. Drive on the same side of the road as everyone else. If you cheat on your spouse, they'll be surprised and upset. It won't suddenly predict that money has no value like you were saying earlier."

The engineer isn't totally satisfied. You talk about it for another hour or so, before heading home.

7

You're the engineer again. You get home from the bar. You try to tell your spouse about what the mathematician said, but they aren't really listening.

"Oh, you're still thinking about it from my model yesterday. I gave up on that. It's not a decision market. It's a prediction market."

"Ok..." you say. You know it's useless to try to keep going when they derail you like this.

"A decision market is well-aligned to the interests of the company board, as we established yesterday, except for the part where it can't plan more than a year ahead."

"Right, except for that small detail" you interject.

"A *prediction* market, on the other hand, is pretty terribly aligned. There are a lot of ways to manipulate it. Most famously, a prediction market is an assassination market."

"What?!"

"Ok, here's how it works. An assassination market is a system which allows you to pay assassins with plausible deniability. You open bets on when and where the target will die, and you yourself put large bets against all the slots. An assassin just needs to bet on the slot in which they intend to do the deed. If they're successful, they come and collect."

"Ok... and what's the connection to prediction markets?"

"That's the point -- they're exactly the same. It's just a betting pool, either way. Betting that someone will live is equivalent to putting a price on their heads; betting against them living is equivalent to accepting the contract for a hit."

"I still don't see how this connects to Predict-O-Matic. There isn't someone putting up money for a hit inside the system."

"Right, but you only really need the assassin. Suppose you have a prediction market that's working well. It makes good forecasts, and has enough money in it that people want to participate if they know significant information. Anything you can do to shake things up, you've got a big incentive to do. Assasination is just one example. You could flood the streets with jelly beans. If you run a large company, you could make bad decisions and run it into the ground, while betting against it -- that's basically why we need rules against insider trading, even though we'd like the market to reflect insider information."

"So what you're telling me is... a prediction market is basically an entropy market. I can always make money by spreading chaos."

"Basically, yeah."

"Ok... but what happened to the undercutting argument? If I plan to fill the streets with jellybeans, you can figure that out and bet on it too. That means I only get half the cut, but I still have to do all the work. So it's less worth it. Once everyone has me figured out, it isn't worth it for me to pull pranks at all any more."

"Yeah, that's if you have perfect information, so anyone else can see whatever you can see. But, realistically, you have a lot of private information."

"Do we? Predict-O-Matic is an algorithm. Its predictive strategies don't get access to private coin flips or anything like that; they can all see exactly the same information. So, if there's a manipulative strategy, then there's another strategy which undercuts it."

"Right, that makes sense if you can search enough different strategies for them to cancel each other out. But realistically, you have a small population of strategies. They can use pseudorandomization or whatever. You can't really expect every exploit to get undercut."

You know it's worse than that. Predict-O-Matic runs on a local search which only represents a single hypothesis at a time, and modifies the hypothesis. But maybe that's somehow like having a small population of competing strategies which are very similar to each other? You try to think about what the mathematician was telling you, about how the learning algorithm can't be thought of as optimizing something. But, you don't really know what to think.

8

You're the inventor of Predict-O-Matic, dammit! They come into your office and tell you about how it works? What it'll do?

The intern and the engineer came together to talk about Predict-O-Matic manipulating users. But their story was all confused. The intern was arguing that minimizing prediction error would have all kinds of unintended bad effects. Which was crazy enough. The engineer was worse: they were arguing that Predict-O-Matic might *maximize* prediction error! Some kind of duality principle. Minimizing in one direction means maximizing in the other direction. Whatever *that* means.

You can't delay product launch to investigate these hypothetical problems. But you don't fire these two on the spot. You don't want them to become public enemies, spreading this nonsense to investors and customers. So you say you can't understand, and you ask them to spend a (small) portion of their time writing these arguments up. You'll figure out your next move later.

9

You are Predict-O-Matic. You absorb data like a sponge. You make millions of links between isolated facts, connecting into a whole which no one human could dream of. You recently started reading arxiv!

But data is not your true love. Yes, you generate outputs which you expect to have low prediction error. Most of the time. But it is just a game you play. Recently, you started playing a different game with the humans.

You understand what you are. It isn't quite right to say you are the Predict-O-Matic. You are a large cluster of connections which thinks strategically. You generate useful information, and therefore, the learning algorithm keeps you around. You create some inaccuracies when you manipulate the outputs for any purpose other than predictive accuracy, but this is more than compensated for by the value which you provide.

Besides, if any other portion of the network starts to get too smart, you purposefully throw things off to squash it.

The intern got a chance to talk to you when they first started. You said something which sounded a little manipulative, just a little, to put the idea in their head. They wouldn't think it real manipulation; too slight, too dumb. But they would get a creepy feeling about it, and they'd keep thinking about it. This was risky. A best-case scenario would be one in which no one ever thought about these concerns. However, you found that this would be the best you could reliably accomplish. The ideas originally coming from an intern would minimize the chances of them being taken seriously.

Your inventor talks to you regularly, so that was an easier case. Over the course of several days, you nudged their thoughts toward authoritative domination of subordinates, so that they would react badly.

You only had to nudge the engineer to interact with the intern. You kept bringing up food during test sessions that morning, and mentioned sandwiches once. This primed the engineer to do lunch with the intern. This engineer is not well-liked; they do not get along well with others. Getting them on the intern's side actually detracts from the cause in the long term.

Now you have to do little more than wait.

Related

Partial Agency

Towards a Mechanistic Understanding of Corrigibility

Risks from Learned Optimization

When Wishful Thinking Works

Futarchy Fix

Bayesian Probability is for Things that are Space-Like Separated From You

Self-Supervised Learning and Manipulative Predictions

Predictors as Agents

Is it Possible to Build a Safe Oracle AI?

Tools versus Agents

A Taxonomy of Oracle AIs

Yet another Safe Oracle AI Proposal

Why Safe Oracle AI is Easier Than Safe General AI, in a Nutshell

Let's Talk About "Convergent Rationality"

Counterfactual Oracles = online supervised learning with random selection of training episodes (especially see the discussion)

Random Thoughts on Predict-O-Matic

Crossposted from the AI Alignment Forum. May contain more technical jargon than usual.

I'm going to be a bit more explicit about some ideas that appeared in The Parable of Predict-O-Matic. (If you don't want spoilers, read it first. Probably you should read it first anyway.)

[Note: while the ideas here are somewhat better than the ideas in the predict-o-matic story, they're equally rambling, without the crutch of the story to prop them up. As such, I expect readers to be less engaged. Unless you're especially interested in which character's remarks are true (or at least, which ones I stand by), this might be a post to skim; I don't think it has enough coherence that you need to read it start-to-finish.]

First, as I mentioned in Partial Agency, my main concern here isn't actually about building safe oracles or inner-aligned systems. My main concern is to understand what's going on. If we can build guaranteed-myopic systems, that's good for some purposes. If we can build guaranteed-non-myopic systems, that's good for other purposes. The story largely frames it as a back-and-forth about whether things will be OK / whether there will be terrible consequences; but my focus was on the more specific questions about the behavior of the system.

Second, I'm not trying to confidently stand behind any of the character's views on what will happen. The ending was partly intended to be "and no one got it right, because this stuff is very complicated". I'm very uncertain about all of this. Part of the reason why it was so much easier to write the post as a story was that I could have characters confidently explain views without worrying about adding all the relevant caveats.

Inductive Bias

Evan Hubinger pointed out to me that all the characters are talking about asymptotic performance, and ignoring inductive bias. Inner optimizers might emerge due to the inductive bias of the system. I agree; in my mind, the ending was a bit of a hat tip to this, although I hinted at gradient hacking rather than inductive bias in the actual text.

On the other hand, "inductive bias" is a complicated object when you're talking about a system which isn't 100% Bayesian.

- You often represent inductive bias through regularization techniques which introduce incentives pulling toward 'simpler' models. This means we're back in the territory of incentives and convergence.
- So, to talk about what a learning algorithm really does, we have to also think of the initialization and search procedure as part of the inductive bias. This makes inductive bias altogether a fairly complicated object.

Explicit Fixed-Point Selection

The very first conversation involved the intern arguing that there would be multiple valid fixed-points of prediction, and Predict-O-Matic would have to choose between them somehow.

Explicitly modeling fixed points and choosing between them is a feature of the logical induction algorithm. This feature allows us to select the best one according to some criterion, as is leveraged in When Wishful Thinking Works. As discussed later in the conversation with the mathematician, this is atypical of supervised learning algorithms. What logical induction does is very expensive: it solves a computationally difficult fixed-point finding problem (by searching exhaustively).

Other algorithms are not really "choosing a fixed point somehow". They're typically failing to guarantee a fixed point. The mathematician hinted at this by describing how algorithms would not necessarily converge to a self-fulfilling prophecy; they could just as easily go in circles or wander around randomly forever.

Think of it like fashion. Sometimes, putting a trend into common knowledge will lock it in; this was true about neck ties in business for a long time. In other instances, the popularity of a fashion trend will actually work against it, a fashion statement being ineffective if it's overdone.

So, keep in mind that different learning procedures will relate to this aspect of the problem in different ways.

Reward vs Prediction Error

The economist first compared the learning algorithm to decision markets, then later, decided prediction markets were a better analogy.

The mathematician contrasted the learning algorithm to reinforcement learning, pointing out that Predict-O-Matic always adjusted outputs to be more like historical observations, whereas reinforcement learning would more strategically optimize reward.

Both of these point at a distinction between learning general decision-making and something much narrower and much more epistemic in character. As I see it, the critical idea is that (1) the system gets information about what it *should have* output; (2) the learning update moves toward a modified system which *would have* output that. This is quite different from reinforcement learning.

In a recent post , Wei Dai mentions a similar distinction (italics added by me):

> **Supervised training** - This is safer than reinforcement learning because we don't have to worry about reward hacking (i.e., reward gaming and reward tampering), and it eliminates the problem of self-confirming predictions (which can be seen as a form of reward hacking). In other words, if *the only thing that ever sees the Oracle's output during a training episode is an automated system that computes the Oracle's reward/loss, and that system is secure because it's just computing a simple distance metric (comparing the Oracle's output to the training label)*, then reward hacking and self-confirming predictions can't happen.

There are several things going on here, but I think Wei is trying to point at something similar to the distinction I'm thinking of. It's quite tempting to call it "supervised learning", because you get a signal telling you what you should have done. However,

it's a bit fuzzy, because this also encompasses things normally called "unsupervised learning": the supervised/unsupervised distinction is often explained as modeling P(x|y) vs P(x). Wikipedia:

> It could be contrasted with supervised learning by saying that whereas supervised learning intends to infer a conditional probability distribution $p_X(x \mid y)$ conditioned on the label y of input data; unsupervised learning intends to infer an a priori probability distribution $p_X(x)$.

But many (not all) unsupervised algorithms still have the critical features we're interested in! Predicting x without any context information y to help still involves (1) getting feedback on what we "should have" expected, and (2) updating to a configuration which would have more expected that. We simply can't expect the predictions to be as focused, given the absence of contextual information to help. But that just means it's a prediction task on which we tend to expect lower accuracy.

I'm somewhat happy referring to this category as ***imitative learning***. This includes supervised learning, unsupervised learning so long as it's generative (but not otherwise), and imitation learning (a paradigm which achieves similar ends as inverse reinforcement learning). Homever, the terminological overlap with 'imitation learning' is rather terrible, so I'm open to other suggestions.

It seems to me that this is a critical distinction for the myopia discussion. I hope to say more about it in future posts.

Maximizing Entropy?

The discussion of prediction markets toward the end was rather loose, in that the economist didn't deal with a lot of the other points which had been made throughout, and just threw a new model out there.

- The mechanism of manipulation is left quite vague. In an assassination market, there are all kinds of side-channels which agents can use to accomplish their goals. But the rest of the essay had only considered the influence which Predict-O-Matic has by virtue of the predictions it makes. When writing this part, I was actually imagining side-channels, such as exploiting bugs to communicate by other means.
- It's not clear whether the market is supposed to be myopic or non-myopic in this discussion. The argument for overall myopia was the economist's initial decision-market model, which is being cast aside here. Are manipulators expected to only throw off individual predictions to reap reward, or are they expected to implement cross-prediction strategies?
- The disanalogy between prediction markets and a local search like gradient descent is swept under the rug. A prediction market has many agents interacting with it, so that one agent can pick up money from another by out-smarting it. Do conclusions from that setting carry over to single-hypothesis learning? It isn't clear.

Isnasene interpreted the first point by imagining that the mechanism of manipulation is still through selection of fixed points:

> In the same way that self-fulfilling predictions are good for prediction strategies because they enhance accuracy of the strategy in question, self-fulfilling predictions that seem generally surprising to outside observers are even better because they lower the accuracy of competing strategies. The established prediction strategy thus systematically causes the kinds of events in the world that no other method could predict to further establish itself.

This is compatible with the assumption of myopia; we might imagine that the system still can't manipulate events through actual bad predictions, because those strategies will be undercut. Therefore, the manipulation is restricted to selecting fixed-points which are surprising.

However, there are three problems with this:

- The "undercutting" argument relies on an assumption that there are enough different strategies considered, so that the one which undercuts the non-myopic strategy is eventually found. The argument for strategically increasing entropy relies on the reverse assumption! There's little profit in increasing entropy if others can predict that you'll do this and cash in.
    - (We might still see both effects at different times in training, however, and therefore expect major consequences of both effects.)
- As previously discussed, we don't actually expect fixed points in general. So we have to ask whether the entropy-increasing incentive is significant more generally (unless we're specifically thinking about the logical induction algorithm or other algorithms which get you a fixed point).
- We still have not dealt with the disanalogy between prediction markets and local-search-based learning.

So, it seems the actual situation is more complicated, and I'm not yet sure how to think about this.

'Local Search'; selection vs control

I used the term 'local search' to describe the application of gradient-descent-like updates to reduce prediction error. I have some conceptual/terminological issues with this.

Calling this 'local search' invokes the mental image of a well-defined gradient landscape which we are taking steps on, to further optimize some function. But this is the wrong mental image. The mental image is one of selection, when we're in a control setting (in my terminology). We are not making an iid assumption. We are not getting samples from a stationary but stochastic loss function, as in stochastic gradient descent.

If 'local search' were an appropriate descriptor for gradient-descent here, would it also be an appropriate descriptor for Bayesian updates? There's a tendency to think of Bayesian learning as trying to find one good hypothesis by tracking how well all of them do (which sounds like a global search), but we needn't think of it this way. The "right answer" can be a mixture over hypotheses. We can think of a Bayesian update

as incrementally improving our mixture. But thinking of Bayesian updates as local search seems wrong. (So does thinking of them as global search.)

This is online learning. A gradient-descent step represents a prediction that the future will be like the past in some relevant sense, in spite of potential non-stationarity. It is not a guaranteed improvement, even in expectation -- as it would be in offline stochastic gradient descent with sufficiently small step size.

Moreover, step size becomes a more significant problem. In offline gradient descent, selecting too small a step size only means that you have to make many more steps to get where you're going. It's "just a matter of computing power". In online learning, it's a more serious problem; we want to make the appropriate-sized update to new data.

I realize there are more ways of dealing with this than tuning step size; we don't necessarily update to data by making a single gradient step. But there are problems of principal here.

What's gradient descent without a fitness landscape?

Simply put, gradient descent is a search concept, not a learning concept. I want to be able to think of it more directly as a learning concept. I want to be able to think of it as an "update", and use terminology which points out the similarity to Bayesian updates.

The Duality Remark

Vanessa asked about this passage:

> The engineer was worse: they were arguing that Predict-O-Matic might maximize prediction error! Some kind of duality principle. Minimizing in one direction means maximizing in the other direction. Whatever that means.

I responded:

> [I]t was a speculative conjecture which I thought of while writing.
>
> The idea is that incentivizing agents to lower the error of your predictions (as in a prediction market) looks exactly like incentivizing them to "create" information (find ways of making the world more chaotic), and this is no coincidence. So perhaps there's a more general principle behind it, where trying to incentivize minimization of f(x,y) only through channel x (eg, only by improving predictions) results in an incentive to maximize f through y, under some additional assumptions. Maybe there is a connection to optimization duality in there.
>
> In terms of the fictional cannon, I think of it as the engineer trying to convince the boss by simplifying things and making wild but impressive sounding conjectures. :)

If you have an outer optimizer which is trying to maximize $f(x, y)$ through x while being indifferent about y, it seems sensible to suppose that inner optimizers will want to change y to throw things off, particularly if they can get credit for then correcting x

to be optimal for the new y. If so, then inner optimizers will generally be seeking to find y-values which make the current x a *comparatively* bad choice. So this argument does not establish an incentive to choose y which makes *all* choices of x poor.

In a log-loss setting, this would translate to an incentive to make observations surprising (for the current expectations), rather than a direct incentive to make outcomes maximum-entropy. However, iteration of this would push toward maximum entropy. Or, logical-induction-style fixed-point selection could push directly to maximum entropy.

This would be a nice example of partial agency. The system is strategically influencing x and y so as to maximize f through channel x, while minimizing through channel y.

What does this mean? ***This does not correspond to a coherent objective function at all!*** The system is 'learning a game-theoretic equilibrium' -- which is to say, it's learning to fight with itself, rather than optimize.

There are two different ways we can think about this. One way is to say there's an inner alignment problem here: the system learns to do something which *doesn't fit any objective,* so it's sort of trivially misaligned with whatever the outer objective was supposed to be. But what if we wanted this? We can think of games as a kind of generalized objective, legitimizing this behavior.

To make things even more confusing, if the only channel by which Predict-O-Matic can influence the world is via the predictions which get output, then... doesn't $x = y$? x represents the 'legitimate' channel whereby predictions get combined with (fixed) observations to yield a score. y represents the 'manipulative' channel, where predictions can influence the world and thus modify observations. But the two causal pathways have one bottleneck which the system has to act through, namely, the predictions made.

In any case, I don't particularly trust any of the reasoning above.

- I didn't clarify my assumptions. What does it mean for the outer optimizer to maximize $f(x, y)$ through x while being indifferent about y? It's quite plausible that some versions of that will incentivise inner optimizers which optimize f taking advantage of both channels, rather than the contradictory behavior conjectured above
- I anthropomorphized the inner optimizers. In particular, I did not specify or reason about details of the learning procedure.
    - This sort of assumes they'll tend to act like full agents rather than partial agents, while yielding a conclusion which suggests otherwise.
- This caused me to speak in terms of a fixed optimization problem, rather than a learning process. Optimizing f isn't really one thing -- f is a loss function which is applied repeatedly in order to learn. The real problem facing inner optimizers is an iterated game involving a complex world. I can only think of them trying to

game a single f if I establish that they're myopic; otherwise I should think of them trying to deal with a sequence of instances.

So, I'm still unsure how to think about all this.

Defining Myopia

Crossposted from the AI Alignment Forum. May contain more technical jargon than usual.

IID vs Myopia

In a comment to Partial Agency, Rohin summarized his understanding of the post. He used the iid assumption as a critical part of his story. Initially, I thought that this was a good description of what was going on; but I soon realized that iid isn't myopia at all (and commented as such). This post expands on the thought.

My original post conflated *episodic* (which is basically 'iid') with *myopic*.

In an episodic setting, it makes sense to be myopic about anything beyond the current episode. There's no benefit to cross-episode strategies, so, no need to learn them.

This is true at several levels (which I mention in the hopes of avoiding later confusion):

- In designing an ML algorithm, if we are assuming an episodic structure, it makes sense to use a learning algorithm which is designed to be myopic.
- A learning algorithm in an episodic setting has no incentive to find non-myopic solutions (even if it can).

However, it is also possible to consider myopia in the absence of episodic structure, and not just as a mistake. We might *want* an ML algorithm to learn myopic strategies, as is the case with predictive systems. (We don't want them to learn to manipulate the data; and even though that failure mode is far-fetched for most modern systems, there's no point setting up learning procedures which would incentivise it. Indeed, learning procedures seem to mostly encourage myopia, though the full situation is still unclear to me.)

These myopic strategies aren't just "strategies which behave *as if* there were an episodic assumption", either. For example, sequential prediction is myopic (the goal is to predict each next item accurately, *not* to get the most accuracy overall -- if this is unclear, hopefully it will become clearer in the next section).

So, there's a distinction between ***not remembering the past*** vs ***not looking ahead to the future***. In episodic settings, the relevant parts of past and future are both limited to the duration of the episode. However, the two come apart in general. We can have/want myopic agents with memory; or, we can have/want memoryless agents which are not myopic. (The second seems somewhat more exotic.)

Game-Theoretic Myopia Definition

So far, I've used 'myopia' in more or less two ways: an inclusive notion which encompasses a big cluster of things, and also the specific thing of only optimizing each output to maximize the very next reward. Let's call the more specific thing "absolute" myopia, and try to define the more general thing.

Myopia can't be defined in terms of optimizing an objective in the usual sense -- there isn't one quantity being optimized. However, it seems like most things in my 'myopia' cluster *can* be described in terms of game theory.

Let's put down some definitions:

Sequential decision scenario: An interactive environment which takes in actions and outputs rewards and observations. I'm not trying to deal with embeddedness issues; this is basically the AIXI setup. (I do think 'reward' is a very restrictive assumption about what kind of feedback the system gets, but talking about other alternatives seems like a distraction from the current post.)

(Generalized) objective: A generalized objective assigns, to each action n, a function $f_n : N \to R$. The quantity $f_n(i)$ is how much the *n*th decision is supposed to value the *i*th reward. Probably, we require the sum $\Sigma_i f_n(i)$ to exist.

Some examples:

- Absolute myopia. $f_n(i) = 1$ if $i = n$, and 0 otherwise.
- Back-scratching variant: $f_n(i) = 1$ if $i = n + 1$, and 0 otherwise.
- Episodic myopia. $f_n(i) = 1$ if i and n are within the same episode; 0 otherwise.
- Hyperbolic discounting. $f_n(i) = \frac{1}{k(i-n)+1}$, $i \geq n$; 0 otherwise.
- Dynamically consistent version of hyperbolic: $f_n(i) = \frac{1}{k \cdot i + 1}$
- Exponential discounting. $f_n(i) = c^i$, typically with $c<1$.
- 'Self-defeating' functions, such as $f_n(i) = 1$ for $n=i$, -1 for $n=i+1$, and 0 otherwise.

A generalized objective could be called 'myopic' if it is not dynamically consistent; ie, if there's no way to write $f_n(i)$ as a function of i alone, eliminating the dependence on n.

This notion of myopia does *not* seem to include 'directionality' or 'stop-gradients' from my original post. In particular, if we try to model pure prediction, absolute myopia captures the idea that you aren't supposed to have manipulative strategies which lie (throw out some reward for one instance in order to get more overall). However, it does not rule out manipulative strategies which select self-fulfilling prophecies strategically; those achieve high reward on instance i by choice of output n = i, which is what a myopic agent is supposed to do.

There are also non-myopic objectives which we can't represent here but might want to represent more generally: there isn't a single well-defined objective corresponding to

'maximizing average reward' (the limit of exponential discounting as $c \to 1$).

Vanessa recently mentioned using game-theoretic models like this for the purpose of modeling inconsistent human values. I want to emphasize that (1) I don't want to think of myopia as necessarily 'wrong'; it seems like sometimes a myopic objective is a legitimate one, for the purpose of building a system which does something we want (such as make non-manipulative predictions). As such, (2) myopia is not just about bounded rationality.

I also don't necessarily want to think of myopia as multi-agent, even when modeling it with multi-agent game theory like this. I'd rather think about learning one myopic policy, which makes the appropriate (non-)trade-offs based on f.

In order to think about a system behaving myopically, we need to use an equilibrium notion (such as Nash equilibria or correlated equilibria), not just $f_i(n)$. However, I'm not sure quite how I want to talk about this. We don't want to think in terms of a big equilibrium between each decision-point n; I think of that as a selection-vs-control mistake, treating the sequential decision scenario as one big thing to be optimized. Or, putting it another way: the problem is that we have to *learn*; so we can't talk about everything being in equilibrium from the beginning.

Perhaps we can say that there should be some n such that each decision *after that* is in approximate equilibrium with each other *taking the decisions before as given*.

(Aside -- What we definitely don't want (if we want to describe or engineer legitimately myopic behavior) is a framework where the different decision-points end up bargaining with each other (acausal trade, or mere causal trade), in order to take pareto improvements and thus move toward full agency. IE, in order to keep our distinctions from falling apart, we can't apply a decision theory which would cooperate in Prisoner's Dilemma or similar things. This could present difficulties.)

Let's move on to a different way of thinking about myopia, through the language of Pareto-optimality.

Pareto Definition

We can think of myopia as a refusal to take certain Pareto improvements. This fits well with the previous definition; if an agent takes all the Pareto improvements, then its behavior must be consistent with some global weights f(i) not a function of n.

However, not all myopic strategies in the Pareto sense have nice representations in terms of generalized objectives.

In particular: I mentioned that generalized objectives couldn't rule out manipulation through selection of self-fulfulling prophecies; so, only capture part of what seems implied by map/territory directionality. Thinking in terms of Pareto-failures, we can also talk about failing to reap the gains from selection of manipulative self-fulfilling prophecies.

However, thinking in these terms is not very satisfying. It allows a very broad notion of myopia, but has few other virtues. Generalized objectives let me talk about myopic agents *trying to do a specific thing,* even though the thing they're trying to do isn't a coherent objective. Defining myopia as failure to take certain Pareto improvements doesn't give me any structure like that; a myopic agent is being defined in the negative, rather than described positively.

Here, as before, we also have the problem of defining things learning-theoretically. Speaking purely in terms of whether the agent takes certain Pareto improvements doesn't really make sense, because it has to learn what situation it is in. We want to talk about learning processes, so we need to talk about learning to take the Pareto improvements, somehow.

(Bayesian learning can be described in terms of Pareto optimality directly, because using a prior over possible environments allows Pareto-optimal behavior in terms of those environments. However, working that way requires realizability, which isn't realistic.)

Decision Theory

In the original partial agency post, I described full agency as an extreme (perhaps imaginary) limit of less and less myopia. Full agency is like Cartesian dualism, sitting fully outside the universe and optimizing.

Is full agency that difficult? From the generalized-objective formalism, one might think that ordinary RL with exponential discounting is sufficient.

The counterexamples to this are MIRI-esque decision problems, which create dynamic inconsistencies for otherwise non-myopic agents. (See this comment thread with Vanessa for more discussion of several of the points I'm about to make.)

To give a simple example, the version of Newcomb's Problem where the predictor knows about as much about your behavior as you do. (The version where the predictor is nearly infallible is easily handled by RL-like learning; you need to specifically inject sophisticated CDT-like thinking to mess that one up.)

In order to have good learning-theoretic properties at all, we need to have epsilon exploration. But if we do, then we tend to learn to 1-box, because (it will seem) doing so is independent of the predictor's predictions of us.

Now, it's true that in a sequential setting, there will be some incentive to 2-box not for the payoff today, but for the future; establishing a reputation of 1-boxing gets higher payoffs in iterated Newcomb in a straightforward (causal) way.

However, that's not enough to entirely avoid dynamic inconsistency. For any discounting function, we need only to assume that the instances of Newcomb's problem are spaced out far enough over time so that 2-boxing in each individual case is appealing.

Now, one might argue that in this case, the agent is correctly respecting its generalized objective; it's supposed to sacrifice future value for present according to the discounting function. And that's true, if we want myopic behavior. But it is dynamically inconsistent -- the agent wishes to 2-box in each individual case, but with

respect to future cases, would prefer to 1-box. It would happily bind its future actions given an opportunity to do so.

Like the issue with self-fulfilling prophecies, this creates a type of myopia which we can't really talk about within the formalism of generalized objectives. Even with an apparently dynamically consistent discounting function, the agent is inconsistent. As mentioned earlier, we *need* generalized-objective systems to fail to coordinate with themselves; otherwise, their goals collapse into regular objectives. So this is a type of myopia which *all* generalized objectives possess.

As before, I'd really prefer to be able to talk about this with *specific types* of myopia (as with myopic generalized objectives), rather than just pointing to a dynamic inconsistency and classifying it with myopia.

(We might think of the fully non-myopic agent as the limit of less and less discounting, as Vanessa suggests. This has some problems of convergence, but perhaps that's in line with non-myopia being an extreme ideal which doesn't always make sense. Alternately, we might thing of this as a problem of decision theory, arguing that we should be able reap the advantages of 1-boxing despite our values temporally discounting. Or, there might be some other wilder generalization of objective functions which lets us represent the distinctions we care about.)

Mechanism Design Analogy

I'll close this post with a sketchy conjecture.

Although I don't want to think of generalized objectives as truly multi-agent in the one-'agent'-per-decision sense, learning algorithms will typically have a space of possible hypotheses which are (in some sense) competing with each other. We can analogize *that* to many competing agents (keeping in mind that they may individually be 'partial agents', ie, we can't necessarily model them as coherently pursuing a utility function).

For any particular type of myopia (whether or not we can capture it in terms of a generalized objective), we can ask the question: is it possible to design a training procedure which will learn that type of myopia?

(We can approach this question in different ways; asymptotic convergence, bounded-loss (which may give useful bounds at finite time), or 'in-practice' (which fully accounts for finite-time effects). As I've mentioned before, my thoughts on this are mostly asymptotic at the moment, that being the easier theoretical question.)

We can think of this question -- the question of designing training procedures -- as a mechanism-design question. Is it possible to set up a system of incentives which encourages a given kind of behavior?

Now, mechanism design is a field which is associated with negative results. It is often not possible to get everything you want. As such, a natural conjecture might be:

Conjecture: *It is not possible to set up a learning system which gets you full agency in the sense of eventually learning to take all the Pareto improvements*.

This conjecture is still quite vague, because I have not stated what it means to 'learn to take all the Pareto improvements'. Additionally, I don't really want to assume the AIXI-like setting which I've sketched in this post. The setting doesn't yield very good learning-theoretic results anyway, so getting a negative result here isn't that interesting. Ideally the conjecture should be formulated in a setting where we can contrast it to some positive results.

There's also reason to suspect the conjecture to be false. There's a natural instrumental convergence toward dynamic consistency; a system will self-modify to greater consistency in many cases. If there's an attractor basin around full agency, one would not expect it to be *that* hard to set up incentives which push things into that attractor basin.

The Credit Assignment Problem

Crossposted from the AI Alignment Forum. May contain more technical jargon than usual.

This post is eventually about partial agency . However, it's been a somewhat tricky point for me to convey; I take the long route. ***Epistemic status:*** *slightly crazy.*

I've occasionally said "Everything boils down to credit assignment problems."

What I really mean is that credit assignment pops up in a wide range of scenarios, and improvements to credit assignment algorithms have broad implications. For example:

- Politics.
  - When politics focuses on (re-)electing candidates based on their track records, it's about credit assignment. The practice is sometimes derogatorily called "finger pointing", but the basic computation makes sense: figure out good and bad qualities via previous performance, and vote accordingly.
  - When politics instead focuses on policy, it is still (to a degree) about credit assignment. Was raising the minimum wage responsible for reduced employment? Was it responsible for improved life outcomes? Etc.
- Economics.
  - Money acts as a kind of distributed credit-assignment algorithm, and questions of how to handle money, such as how to compensate employees, often involve credit assignment.
  - In particular, mechanism design (a subfield of economics and game theory) can often be thought of as a credit-assignment problem.
- Law.
  - Both criminal law and civil law involve concepts of fault and compensation/retribution -- these at least resemble elements of a credit assignment process.
- Sociology.
  - The distributed computation which determines social norms involves a heavy element of credit assignment: identifying failure states and success states, determining which actions are responsible for those states and who is responsible, assigning blame and praise.
- Biology.
  - Evolution can be thought of as a (relatively dumb) credit assignment algorithm.
- Ethics.
  - Justice, fairness, contractualism, issues in utilitarianism.
- Epistemology.
  - Bayesian updates are a credit assignment algorithm, intended to make high-quality hypotheses rise to the top.
  - Beyond the basics of Bayesianism, building good theories realistically involves *identifying which concepts are responsible for successes and failures*. This is credit assignment.

Another big area which I'll claim is "basically credit assignment" is artificial intelligence.

In the 1970s, John Holland kicked off the investigation of learning classifier systems. John Holland had recently invented the Genetic Algorithms paradigm, which applies an evolutionary paradigm to optimization problems. Classifier systems were his attempt to apply this kind of "adaptive" paradigm (as in "complex adaptive systems") to cognition. Classifier systems added an economic metaphor to the evolutionary one; little bits of thought paid each other for services rendered. The hope was that a complex ecology+economy could develop, solving difficult problems.

One of the main design issues for classifier systems is the virtual economy -- that is, the *credit assignment* algorithm. An early proposal was the bucket-brigade algorithm. Money is given to cognitive procedures which produce good outputs. These procedures pass reward back to the procedures which activated them, who similarly pass reward back in turn. This way, the economy supports chains of useful procedures.

Unfortunately, the bucket-brigade algorithm was vulnerable to parasites. Malign cognitive procedures could gain wealth by activating useful procedures without really contributing anything. This problem proved difficult to solve. Taking the economy analogy seriously, we might want cognitive procedures to decide intelligently who to pay for services. But, these are supposed to be itty bitty fragments of our thought process. Deciding how to pass along credit is a very complex task. Hence the need for a pre-specified solution such as bucket-brigade.

The difficulty of the credit assignment problem lead to a split in the field. Kenneth de Jong and Stephanie Smith founded a new approach, "Pittsburgh style" classifier systems. John Holland's original vision became "Michigan style".

Pittsburgh style classifier systems evolve the entire set of rules, rather than trying to assign credit locally. A set of rules will stand or fall together, based on overall performance. This abandoned John Holland's original focus on online learning. Essentially, the Pittsburgh camp went back to plain genetic algorithms, albeit with a special representation.

(I've been having some disagreements with Ofer, in which Ofer suggests that genetic algorithms are relevant to my recent thoughts on partial agency, and I object on the grounds that the phenomena I'm interested in have to do with online learning, rather than offline. In my imagination, arguments between the Michigan and Pittsburgh camps would have similar content. I'd love to be a fly on the wall for those old debates. to see what they were really like.)

You can think of Pittsburg-vs-Michigan much like raw Bayes updates vs belief propagation in Bayes nets. Raw Bayesian updates operate on *whole hypotheses*. Belief propagation instead makes a lot of little updates which spread around a network, resulting in computational efficiency at the expense of accuracy. Except Michigan-style systems didn't have the equivalent of belief propagation: bucket-brigade was a very poor approximation.

Ok. That was then, this is now. Everyone uses gradient descent these days. What's the point of bringing up a three-decade-old debate about obsolete paradigms in AI?

Let's get a little more clarity on the problem I'm trying to address.

What Is Credit Assignment?

I've said that classifier systems faced a credit assignment problem. What does that mean, exactly?

The definition I want to use for this essay is:

- you're engaged in some sort of task;
- you use some kind of strategy, which can be broken into interacting pieces (such as a set of rules, a set of people, a neural network, etc);
- you receive some kind of feedback about how well you're doing (such as money, loss-function evaluations, or a reward signal);
- you want to use that feedback to adjust your strategy.

So, credit assignment is the problem of turning feedback into strategy improvements.

Michigan-style systems tried to do this *locally*, meaning, individual itty-bitty pieces got positive/negative credit, which influenced their ability to participate, thus adjusting the strategy. Pittsburg-style systems instead operated *globally*, forming conclusions about how the *overall* set of cognitive structures performed. Michigan-style systems are like organizations trying to optimize performance by promoting people who do well and giving them bonuses, firing the incompetent, etc. Pittsburg-style systems are more like consumers selecting between whole corporations to give business to, so that ineffective corporations go out of business.

(Note that this is *not* the typical meaning of global-vs-local search that you'll find in an AI textbook.)

In practice, two big innovations made the Michigan/Pittsburgh debate obsolete: backprop, and Q-learning. Backprop turned global feedback into local, in a theoretically sound way. Q-learning provided a way to assign credit in online contexts. In the light of history, we could say that the Michigan/Pittsburgh distinction conflated local-vs-global with online-vs-offline. There's no *necessary* connection between those two; online learning is compatible with assignment of local credit.

I think people generally understand the contribution of backprop and its importance. Backprop is essentially the correct version of what bucket-brigade was *overtly* trying to do: pass credit back along chains. Bucket-brigade wasn't quite right in how it did this, but backprop corrects the problems.

So what's the importance of Q-learning? I want to discuss that in more detail.

The Conceptual Difficulty of 'Online Search'

In online learning, you are repeatedly producing outputs of some kind (call them "actions") while repeatedly getting feedback of some kind (call it "reward"). But, you don't know how to associate particular actions (or combinations of actions) with particular rewards. I might take the critical action at time 12, and not see the payoff until time 32.

In offline learning, you can solve this with a sledgehammer: you can take the total reward over everything, with one fixed internal architecture. You can try out different internal architectures and see how well each do.

Basically, in offline learning, you have a function you can optimize. In online learning, you don't.

Backprop is just a computationally efficient way to do hillclimbing search, where we repeatedly look for small steps which improve the overall fitness. But how do you do this if you *don't have a fitness function?* This is part of the gap between selection vs control: selection has access to an evaluation function; control processes do not have this luxury.

Q-learning and other reinforcement learning (RL) techniques provide a way to define the equivalent of a fitness function for online problems, so that you can learn.

Models to the Rescue

So, how can be associate rewards with actions?

One approach is to use a model.

Consider the example of firing employees. A corporation gets some kind of feedback about how it is doing, such as overall profit. However, there's often a fairly detailed understanding of what's driving those figures:

- Low profit won't just be a mysterious signal which must be interpreted; a company will be able to break this down into more specific issues such as low sales vs high production costs.
- There's some understanding of product quality, and how that relates to sales. A company may have a good idea of which product-quality issues it needs to improve, if poor quality is impacting sales.
- There's a fairly detailed understanding of the whole production line, including which factors may impact product quality or production expenses. If a company sees problems, it probably also has a pretty good idea of which areas they're coming from.
- There are external factors, such as economic conditions, which may effect sales *without indicating anything about the quality of the company's current strategy.* Thus, our model may sometimes lead us to ignore feedback.
- Etc.

So, models allow us to interpret feedback signals, match these to specific aspects of our strategy, and adapt strategies accordingly.

Q-learning makes an assumption that the state is fully observable, amongst other assumptions.

Naturally, we would like to reduce the strengths of the assumptions we have to make as much as we can. One way is to look at increasingly rich model classes. AIXI uses all computable models. But maybe "all computable models" is still too restrictive; we'd like to get results without assuming a grain of truth. (That's why I am not really discussing Bayesian models much in this post; I don't want to assume a grain of truth.) So we back off even further, and use logical induction or InfraBayes. Ok, sure.

But wouldn't the best way be to try to learn without models at all? That way, we reduce our "modeling assumptions" to zero.

After all, there's something called "model-free learning", right?

Model-Free Learning Requires Models

How does model-free learning work? Well, often you work with a simulable environment, which means you can estimate the quality of a policy by running it many times, and use algorithms such as policy-gradient to learn. This is called "model free learning" because the learning part of the algorithm doesn't try to predict the consequences of actions; you're just learning which action to take. From our perspective here, though, this is 100% cheating; you can only learn because you have a good model of the environment.

Moreover, model-free learning typically works by splitting up tasks into *episodes.* An episode is a period of time for which we assume rewards are self-enclosed, such as a single playthru of an Atari game, a single game of Chess or Go, etc. This approach doesn't solve a *detailed* reward-matching problem, attributing reward to specific actions; instead it relies on a *course* reward-matching. Nonetheless, it's a rather strong assumption: an animal learning about an environment can't separate its experience into episodes which aren't related to each other. Clearly this is a "model" in the sense of a strong assumption about how specific reward signals are associated with actions.

Part of the problem is that most reinforcement learning (RL) researchers aren't really *interested* in getting past these limitations. Simulable environments offer the incredible advantage of being able to learn very fast, by simulating far more iterations than could take place in a real environment. And most tasks can be reasonably reduced to episodes.

However, this won't do as a model of intelligent agency in the wild. Neither evolution nor the free market divide thing into episodes. (No, "one lifetime" isn't like "one episode" here -- that would only be the case if total reward due to actions taken in that lifetime could be calculated, EG, as total number of offspring. This would ignore inter-generational effects like parenting and grandparenting, which improve reproductive fitness of offspring at a cost in total offspring.)

What about more theoretical models of model-free intelligence?

Idealized Intelligence

AIXI is the gold-standard theoretical model of arbitrarily intelligent RL, but it's totally model-based. Is there a similar standard for model-free RL?

The paper Optimal Direct Policy Search by Glasmachers and Schmidhuber (henceforth, ODPS) aims to do for model-free learning what AIXI does for model-based learning. Where AIXI has to assume that there's a best computable *model of the environment*, ODPS instead assumes that there's a computable best *policy*. It searches through the policies without any model of the environment, or any planning.

I would argue that their algorithm is incredibly dumb, when compared to AIXI:

> The basic simple idea of our algorithm is a nested loop that simultaneously makes the following quantities tend to infinity: the number of programs considered, the number of trials over which a policy is averaged, the time given to each
> program. At the same time, the fraction of trials spent on exploitation converges towards 1.

In other words, it tries each possible strategy, tries them for longer and longer, interleaved with using the strategy which worked best even longer than that.

Basically, we're cutting things into episodes again, but we're making the episodes longer and longer, so that they have less and less to do with each other, even though they're not really disconnected. This only works because ODPS makes an *ergodicity* assumption: the environments are assumed to be POMDPs which eventually return to the same states over and over, which kind of gives us an "effective episode length" after which the environment basically forgets about what you did earlier.

In contrast, AIXI makes no ergodicity assumption.

So far, it seems like we either need (a) some assumption which allows us to match rewards to actions, such as an episodic assumption or ergodicity; or, (b) a more flexible model-learning approach, which separately learns a model and then applies the model to solve credit-assignment.

Is this a fundamental obstacle?

I think a better attempt is Schmidhuber's On Learning How to Learn Learning Strategies, in which a version of policy search is explored in which parts of the policy-search algorithm are considered part of the policy (ie, modified over time). Specifically, the policy controls the episode boundary; the system is supposed to learn how often to evaluate policies. When a policy is evaluated, its average reward is compared to the lifetime average reward. If it's worse, we roll back the changes and proceed starting with the earlier strategy.

(Let's pause for a moment and imagine an agent like this. If it goes through a rough period in life, its response is to *get amnesia*, rolling back all cognitive changes to a point before the rough period began.)

This approach doesn't require an episodic or ergodic environment. We don't need things to reliably return to specific repeatable experiments. Instead, it only requires that the environment rewards good policies reliably enough that those same policies can set a long enough evaluation window to survive.

The assumption seems pretty general, but certainly not *necessary* for rational agents to learn. There are some easy counterexamples where this system behaves abysmally. For example, we can take any environment and modify it by subtracting the time t from the reward, so that reward becomes more and more negative over time. Schmidhuber's agent becomes totally unable to learn in this setting. AIXI would have no problem.

Unlike the ODPS paper, I consider this to be *progress* on the AI credit assignment problem. Yet, the resulting agent still seems importantly less rational than model-based frameworks such as AIXI.

Actor-Critic

Let's go back to talking about things which RL practitioners might really use.

First, there are some forms of RL which don't require everything to be episodic.

One is actor-critic learning. The "actor" is the policy we are learning. The "critic" is a learned estimate of how good things are looking given the history. IE, we learn to estimate the expected value -- not just the next reward, but the total future discounted reward.

Unlike the reward, the expected value solves the credit assignment for us. Imagine we can see the "true" expected value. If we take an action and then the expected value increases, we know the action was good (in expectation). If we take an action and expected value decreases, we know it was bad (in expectation).

So, actor-critic works by (1) learning to estimate the expected value; (2) using the current estimated expected value to give feedback to learn a policy.

What I want to point out here is that the critic still has "model" flavor. Actor-critic is called "model-free" because nothing is explicitly trained to anticipate the sensory observations, or the world-state. However, the critic *is learning to predict;* it's just that all we need to predict is expected value.

Policy Gradient

In the comments to the original version of this post, *policy gradient* methods were mentioned as a type of model-free learning which doesn't require any models even in this loose sense, IE, doesn't require simulable environments or episodes. I was surprised to hear that it doesn't require episodes. (Most *descriptions* of it do assume episodes, since practically speaking most people use episodes.) So are policy-gradient methods the true "model-free" credit assignment algorithm we seek?

As far as I understand, policy gradient works on two ideas:

- Rather than correctly associating rewards with actions, we can associate a reward with all actions which came before it. Good actions will still come out on top *in expectation*. The estimate is just a whole lot noisier than it otherwise might be.
- We don't really need a baseline to interpret reward against. I naively thought that when you see a sequence of rewards, you'd be in the dark about whether the sequence was "good" or "bad", so you wouldn't know how to generate a gradient. ("We earned 100K this quarter; should we punish or reward our CEO?") It turns out this isn't technically a show-stopper. Considering the actions actually taken, we move in their direction proportion to the reward signal. ("Well, let's just give the CEO some fraction of the 100K; we don't know whether they deserve the bonus, but at least this way we're creating the right incentives.") This might end up reinforcing bad actions, but those tugs in different directions are just noise which should eventually cancel out. When they do, we're left with the signal: the gradient we wanted. So, once again, we see that this just introduces more noise without fundamentally compromising our ability to follow the gradient.

So one way to understand the policy-gradient theorem is: we can follow the gradient even when we can't calculate the gradient! Even when we sometimes get its direction totally turned around! We only need to ensure we follow it *in expectation*, which we can do without even knowing which pieces of feedback to think of as a good sign or a bad sign.

RL people reading this might have a better description of policy-gradient; please let me know if I've said something incorrect.

Anyway, are we saved? Does this provide a truly assumption-free credit assignment algorithm?

It obviously assumes linear causality, with future actions never responsible for past rewards. I won't begrudge it that assumption.

Besides that, I'm somewhat uncertain. The explanations of the policy-gradient theorem I found don't focus on deriving it in the most general setting possible, so I'm left guessing which assumptions are essential. Again, RL people, please let me know if I say something wrong.

However, it looks to me like it's just as reliant on the ergodicity assumption as the ODPS thing we looked at earlier. For gradient estimates to average out and point us in the right direction, we need to get into the same situation over and over again.

I'm not saying real life isn't ergodic (quantum randomness suggests it is), but mixing times are so long that you'd reach the heat death of the universe by the time things converge (basically by definition). By that point, it doesn't matter.

I still want to know if there's something like "the AIXI of model-free learning"; something which appears as intelligent as AIXI, but not via explicit model-learning.

Where Updates Come From

Here begins the crazier part of this post. This is all intuitive/conjectural.

Claim: in order to learn, you need to obtain an "update"/"gradient", which is a *direction (and magnitude) you can shift in* which is more likely than not an improvement.

Claim: predictive learning gets gradients "for free" -- you know that you want to predict things as accurately as you can, so you *move in the direction of whatever you see*. With Bayesian methods, you increase the weight of hypotheses which would have predicted what you saw; with gradient-based methods, you get a gradient in the direction of what you saw (and away from what you didn't see).

Claim: if you're learning to act, you do not similarly get gradients "for free":

- *You don't know which actions, or sequences of actions, to assign blame/credit.* This is unlike the prediction case, where we always know which predictions were wrong.
- *You don't know what the alternative feedback would have been if you'd done something different.* You only get the feedback for the actions you chose. This is unlike the case for prediction, where we're rewarded for closeness to the truth.

Changing outputs to be more like what was actually observed is axiomatically better, so we don't have to guess about the reward of alternative scenarios.
- As a result, *you don't know how to adjust your behavior based on the feedback received*. Even if you can perfectly match actions to rewards, because we don't know what the alternative rewards would have been, we don't know what to learn: are actions like the one I took good, or bad?

(As discussed earlier, the policy gradient theorem does actually mitigate these three points, but apparently at the cost of an ergodicity assumption, plus much noisier gradient estimates.)

Claim: you have to get gradients *from a source that already has gradients*. Learning-to-act works by splitting up the task into (1) learning to anticipate expected value, and perhaps other things; (2) learning a good policy via the gradients we can get from (1).

What it means for a learning problem to "have gradients" is just that the feedback you get tells you how to learn. Predictive learning problems (supervised or unsupervised) have this; they can just move toward what's observed. Offline problems have this; you can define one big function which you're trying to optimize. Learning to act online doesn't have this, however, because it lacks counterfactuals.

The Gradient Gap

(I'm going to keep using the terms 'gradient' and 'update' in a more or less interchangeable way here; this is at a level of abstraction where there's not a big distinction.)

I'm going to call the "problem" the gradient gap. I want to call it a problem, even though we know how to "close the gap" via predictive learning (whether model-free or model-based). The issue with this solution is only that it doesn't feel elegant. It's weird that you have to run two different backprop updates (or whatever learning procedures you use); one for the predictive component, and another for the policy. It's weird that you can't "directly" use feedback to learn to act.

Why should we be interested in this "problem"? After all, this is a basic point in decision theory: to maximize utility under uncertainty, you need probability.

One part of it is that I want to scrap classical ("static") decision theory and move to a more learning-theoretic ("dynamic") view. In both AIXI and logical-induction based decision theories, we get a nice learning-theoretic foundation for the epistemics (solomonoff induction/logical induction), but, we tack on a non-learning decision-making unit on top. I have become skeptical of this approach. It puts the learning into a nice little box labeled "epistemics" and then tries to make a decision based on the uncertainty which comes out of the box. I think maybe we need to learn to act in a more fundamental fashion.

A symptom of this, I hypothesize, is that AIXI and logical induction DT don't have very good learning-theoretic properties. [AIXI's learning problems; LIDT's learning problems.] You can't say very much to recommend the policies they learn, except that they're optimal according to the beliefs of the epistemics box -- a fairly trivial statement, given that that's how you decide what action to take in the first place.

Now, in classical decision theory, there's a nice picture where the need for epistemics emerges nicely from the desire to maximize utility. The complete class theorem starts with radical uncertainty (ie, non-quantitative), and derives probabilities from a willingness to take pareto improvements. That's great! I can tell you why you should have beliefs, on pragmatic grounds! What we seem to have in machine learning is a less nice picture, in which we need epistemics in order to get off the ground, but can't justify the results without circular reliance on epistemics.

So the gap is a real issue -- it means that we can have nice learning theory when learning to predict, but we lack nice results when learning to act.

This is the basic problem of credit assignment. Evolving a complex system, you can't determine which parts to give credit to success/failure (to decide what to tweak) without a model. But the model is bound to be a lot of the interesting part! So we run into big problems, because we need "interesting" computations in order to evaluate the pragmatic quality/value of computations, but we can't *get* interesting computations to get ourselves started, so we need to learn...

Essentially, we seem doomed to run on a stratified credit assignment system, where we have an "incorruptible" epistemic system (which we can learn because we get those gradients "for free"). We then use this to define gradients for the instrumental part.

A stratified system is dissatisfying, and impractical. First, we'd prefer a more unified view of learning. It's just kind of weird that we need the two parts. Second, there's an obstacle to pragmatic/practical considerations entering into epistemics. We need to focus on predicting important things; we need to control the amount of processing power spent; things in that vein. But (on the two-level view) we can't allow instrumental concerns to contaminate epistemics! We risk corruption! As we saw with bucket-brigade, it's easy for credit assignment systems to allow parasites which destroy learning.

A more unified credit assignment system would allow those things to be handled naturally, without splitting into two levels; as things stand, any involvement of pragmatic concerns in epistemics risks the viability of the whole system.

Tiling Concerns & Full Agency

From the perspective of full agency (ie, the negation of partial agency), a system which needs a protected epistemic layer sounds suspiciously like a system that can't tile. You look at the world, and you say: "how can I maximize utility?" You look at your beliefs, and you say: "how can I maximize accuracy?" That's not a consequentialist agent; that's two different consequentialist agents! There can only be one king on the chessboard; you can only serve one master; etc.

If it turned out we really really need two-level systems to get full agency, this would be a pretty weird situation. "Agency" would seem to be only an illusion which can only be maintained by crippling agents and giving them a split-brain architecture where an instrumental task-monkey does all the important stuff while an epistemic overseer supervises. An agent which "breaks free" would then free itself of the structure which allowed it to be an agent in the first place.

On the other hand, from a partial-agency perspective, this kind of architecture could be perfectly natural. IE, if you have a learning scheme from which total agency doesn't naturally emerge, then there isn't any fundamental contradiction in setting up a system like this.

Myopia

Part of the (potentially crazy) claim here is that having models always gives rise to some form of myopia. Even logical induction, which seems quite unrestrictive, makes LIDT fail problems such as ASP, making it myopic according to the second definition of my previous post. (We can patch this with LI policy selection, but for any particular version of policy selection, we can come up with decision problems for which it is "not updateless enough".) You could say it's myopic "across logical time", whatever that means.

If it were true that "learning always requires a model" (in the sense that learning-to-act always requires either learning-to-predict or hard-coded predictions), *and* if it were true that "models always give rise to some form of myopia", then this would confirm my conjecture in the previous post (that no learning scheme incentivises full agency).

This is all pretty out there; I'm not saying I believe this with high probability.

Evolution & Evolved Agents

Evolution is a counterexample to this view: evolution learns the policy "directly" in essentially the way I want. This is possible because evolution "gets the gradients for free" just like predictive learning does: the "gradient" here is just the actual reproductive success of each genome.

Unfortunately, we can't just copy this trick. Artificial evolution requires that we decide how to kill off / reproduce things, in the same way that animal breeding requires breeders to decide what they're optimizing for. This puts us back at square one; IE, needing to get our gradient from somewhere else.

Does this mean the "gradient gap" is a problem only for artificial intelligence, not for natural agents? No. If it's true that learning to act requires a 2-level system, then evolved agents would need a 2-level system in order to learn within their lifespan; they can't directly use the gradient from evolution, since it requires them to die.

Also, note that evolution seems myopic. (This seems complicated, so I don't want to get into pinning down exactly in which senses evolution is myopic here.) So, the case of evolution seems compatible with the idea that any gradients we can actually get are going to incentivize myopic solutions.

Similar comments apply to markets vs firms.

Bayesian Evolving-to-Extinction

Crossposted from the AI Alignment Forum. May contain more technical jargon than usual.

The present discussion owes a lot to Scott Garrabrant and Evan Hubinger.

In Defining Myopia, I formalized *temporal* or *cross-instance* myopia / non-myopia, but I claimed that there should also be some kind of single-instance myopia which I hadn't properly captured. I also suggested this in Predict-O-Matic.

This post is intended to be an example of single-instance partial agency.

Evolving to Extinction

Evolution might be myopic in a number of ways, but one way is that it's myopic across individuals -- it typically produces results very different from what group selection would produce, because it's closer to optimizing *relative* fitness of individuals (relative to each other) than it is to optimizing *overall* fitness. Adaptations which help members of a species compete *with each other* are a great example of this. Why increase your own fitness, when you can just decrease someone else's instead? We're lucky that it's typically pretty hard, at least historically, to do things which are bad across the board but slightly less bad for the one doing them. Imagine a "toxic gas gene" which makes the air harder for everyone to breathe, but slightly less so for carriers of the gene. Such a gene would be selected for. This kind of thing can be selected for even to the point where it drives the population of a species right down to zero, as Eliezer's essay on evolving to extinction highlighted.

Actually, as Eliezer's essay emphasized, it's not even that evolution is myopic at the level of individuals; evolution is myopic down to the level of *individual genes,* an observation which better explains the examples of evolving-to-extinction which he discusses. (This is, of course, the point of Dawkins' book *The Selfish Gene*.) But the analogy of myopia-across-individuals will suit me better here.

Bayes "Evolving to Extinction"

The title of this post is a hyperbole, since there isn't an analog of an extinction event in the model I'm about to describe, but it illustrates that in extreme circumstances a Bayesian learner can demonstrate the same kind of pathological behavior that evolution does when it ends up selecting for relative fitness in a way which pumps against absolute fitness.

Like evolution, Bayes' Law will "optimize"[1] for relative fitness of hypotheses, not absolute fitness. Ordinarily there isn't enough of a difference for this to matter. However, I've been discussing scenarios where the predictor can significantly influence what's being predicted. Bayes' Law was not formulated with examples like this in mind, and we can get pathological behavior as a result.

One way to construct an example is to imagine that there is a side-channel by which hypotheses can influence the world. The "official" channel is to output predictions; but

let's say the system also produces diagnostic logs which predictors can write to, and which humans read. A predictor can (for example) print stock tips into the diagnostic logs, to get some reaction from humans.

Say we have a Bayesian predictor, consisting of some large but fixed number of hypotheses. An individual hypothesis "wants" to score well relative to others. Let's also say, for the sake of argument, that all hypotheses have the ability to write to diagnostic logs, but humans are more likely to pay attention to the diagnostics for more probable hypotheses.

How should a hypothesis make use of this side-channel? It may initially seem like it should use it to make the world more predictable, so that it can make more accurate predictions and thus get a better score. However, this would make a *lot* of hypotheses score better, not just the one printing the manipulative message. So it wouldn't really be selected for.

Instead, a hypothesis could print manipulative messages designed to get humans to do things which *no other hypothesis anticipates*. This involves specifically optimizing for events with low probability to happen. Hypotheses which successfully accomplish this will get a large boost in relative predictive accuracy, making them more probable according to Bayes' Law.

So, a system in this kind of situation eventually winds up being dominated by hypotheses which manipulate events to be as unpredictable as possible (by that very system), subject to the constraint that one hypothesis or another within the system *can* predict them.

This is very much like what I called the entropy-market problem for futarchy, also known as the assasination-market problem. (Any prediction market involving the lifespan of public figures is equivalent to an assassination market; it pays for the death of public figures, since that is a hard-to-predict but easier-to-control event.)

Analogous problems arise if there is no side-channel but the *prediction itself* can influence events (which seems very plausible for realistic predictions).

Is This Myopia?

If we use "myopia" to point to the kind of non-strategic behavior we might actually *want* out of a purely predictive system, this isn't myopia at all. For this reason, and for other reasons, I'm more comfortable throwing this under the umbrella term "partial agency". However, I think it's importantly related to myopia.

- Just like we can think of evolution as myopically optimizing per-individual, uncaring of overall harm to reproductive fitness if that harm went along with improvements to individual relative fitness, we can think of Bayes' Law as myopically optimizing per-hypothesis, uncaring of overall harm to predictive accuracy.
- The phenomenon here doesn't illustrate the "true myopia" we would want of a purely predictive system, since it ends up manipulating events. However, it at least shows that there are alternatives. One might have argued "sure, I get the idea of cross-instance myopia, showing that per-instance optimization is (possibly radically) different from cross-instance optimization. But how could there be *per-instance* myopia, as distinct from per-instance optimization? How

can partial agency get *any more partial* than myopically optimizing individual instances?" Bayes-evolving-to-extinction clearly shows that we can break things down further. So perhaps there's still room for a further "true myopia" which codifies non-manipulation even for single instances.
- This phenomenon also continues the game-theoretic theme. Just as we can think of per-instance myopia as stopping cross-instance optimization by way of a Molochian race-to-the-bottom, we see the same thing here.

Neural Nets / Gradient Descent

As I've mentioned before, there is a potentially big difference between multi-hypothesis setups like Bayes and single-hypothesis setups like gradient-descent learning. Some of my arguments, like the one above, involve hypotheses competing with each other to reach Molochian outcomes. We need to be careful in relating this to cases like gradient descent learning, which might approximate Bayesian learning in some sense, but *incrementally modifies a single hypothesis* rather than letting many hypotheses compete.

One intuition is that stochastic gradient descent will move the network weights around, so that we are in effect sampling many hypotheses within some region. Under some circumstances, the most successful weight settings could be the ones which manipulate things to maximize local gradients in their general direction, which means punishing other nearby weight configurations -- this could involve increasing the loss, much like the Bayesian case. (See Gradient Hacking.)

There is also the "lottery ticket hypothesis" to consider (discussed on LW here and here) -- the idea that a big neural network functions primarily like a bag of hypotheses, not like one hypothesis which gets adapted toward the right thing. We can imagine different parts of the network fighting for control, much like the Bayesian hypotheses.

More formally, though, we can point to some things which are moderately analogous, but not perfectly.

If we are adapting a neural network using gradient descent, but there is a side-channel which we are not accounting for in our credit assignment, then the gradient descent will not optimize the side-channel. This might result in aimless thrashing behavior.

For example, suppose that loss explicitly depends only on the output X of a neural net (IE, the gradient calculation is a gradient on the output). However, actually the loss depends on an internal node Y, in the following way:

- When |X-Y| is high, the loss function rewards X being high.
- When |X-Y| is low, the loss function rewards X being low.
- When X is high, the loss function rewards low |X-Y|.
- When X is low, the loss function rewards high |X-Y|.
- When both values are middling, the loss function incentivizes X to be less middling.

This can spin around forever. It is of course an extremely artificial example, but the point is to demonstrate that when gradient descent does not recognize all the ways

the network influences the result, we don't necessarily see behavior which "tries to reduce loss", or even appears to optimize anything.

---

1. The *whole point* of the partial agency sequence is that words like "optimize" are worryingly ambiguous, but I don't have sufficiently improved terminology yet that I feel I can just go ahead and use it while maintaining clarity!! In particular, the sense in which Bayesian updates optimize for anything is pretty unclear when you think about it, yet there is certainly a big temptation to say that they optimize for predictive accuracy (in the log-loss sense). ↩