My AI Risk Model

Why I'm Worried About AI

Preamble

In this sequence of posts, I want to lay out why I am worried about risks from powerful AI and where I think the specific dangers come from. In general, I think it's good for people to be able to form their own inside views of what’s going on, and not just defer to people. There are surprisingly few descriptions of actual risk models written down.

I think writing down your own version of the AI risk story is good for a few reasons

- It makes you critically examine the risk models of other people, and work out *what you actually believe.*
- It may be helpful for finding research directions which seem good. Solving a problem seems much more doable if you can actually point at it.
- It seems virtuous to attempt to form your own views rather than just the consensus view (whatever that is).
- People can comment on where your story may be weak or inconsistent, which will hopefully push towards the truth.

Additionally, I have some friends and family who are not in EA/AI-safety/Longtermism/Rationality, and it would be nice to be able to point them at something describing why I’m doing what I’m doing (technical AI-safety). Although, admittedly my views are more complicated than I initially thought, so this isn’t a great first introduction to AI-risk.

I don’t expect much or any of these posts to be original and there will be many missing links and references. This can maybe be viewed as a more quick and dirty AGI safety from first principles, with less big picture justification and more focus on my specific risk models. In general, I am concerned most about deceptively aligned AI systems, as discussed in Risks from Learned Optimization.

How do we train neural networks?

In the current paradigm of AI, we train neural networks to be good at tasks, and then we deploy them in the real world to perform those tasks.

We train neural networks on a *training distribution*

- For image classifiers, this is a set of labeled images. E.g. a set of images of cats and dogs with corresponding labels.
- For language models like GPT-3, this is a very large amount of text from the internet.
- For reinforcement learning, this is some training environment where the AI agent can ‘learn’ to take actions. For example, balancing a pole on a cart or playing the Atari game Breakout.

We start with an untrained AI system and modify it to perform better on the training distribution. For our cat vs dog classifier, we feed in images of cats and dogs (our training data) and modify the AI so that it is able to accurately label these images. For GPT-3, we feed in the start of a string of text and modify it such that it can accurately predict what will come next. For our reinforcement learning agent playing Breakout, the agent takes actions in the game (move the platform left and right), we reward actions which lead to high score, and use this reward to train the agent to do well in this game.

If the training is good then our AI system will then be able to generalize and perform well in slightly different domains.

- Our cat vs dog classifier can correctly label images of cats and dogs it hasn't seen before
- Our language model can 'predict' the next word for a given sensible prompt, and use this to generate coherent text
- Our reinforcement learning agent can play the game, even if it hasn't seen this exact game state before. Or even generalize to slightly different environments; what if there are more or differently positioned blocks?

This ability to generalize is limited by the task it was trained on; we can only really expect good generalization on data that is *similar enough* to the training distribution. Our cat vs dog classifier might struggle if we show it a lion. If our language model has only been trained on English and German text, it won't be able to generate French. Our Breakout agent can really only play Breakout.

Optimization

When we train our AI systems we are optimizing them to perform well on the training distribution. By *optimizing* I mean that we are modifying these systems such that they do well on some objective.

Optimized vs Optimizers

It is important to make a distinction between something which is *optimized* and something which is an *optimizer*. When we train our AI systems, we end up with an optimized system; the system has been optimized to perform well on a task, be that cat vs dog classification, predicting the next word in a sentence, or achieving a high score at Breakout. These systems have been *optimized* to do well on the objective we have given them, but they themselves (probably) aren't *optimizers*; they don't have any notion of improving on an objective.

Our cat vs dog classifier likely just has a bunch of heuristics which influence the relative likelihood of 'cat' or 'dog'. Our Breakout agent is probably running an algorithm which looks like "The ball is at position X, the platform is at position Y, so take action A", and not something like "The ball is at position X, the platform is at position Y, if I take action A it will give me a better score than action B, so take action A".

We did the optimizing with our training and ended up with an optimized system.

However, there are reasons to expect that we will get 'optimizers' as we build more powerful systems which operate in complex environments. AI systems can solve a task in 2 main ways (although the boundary here is fuzzy)

- They can use a bunch of heuristics that mechanistically combine to choose an action. For example, “If there is <this texture> +3 to the dog number, if there is <this triangle shape> +4 to the cat number, if there is <this color> next to <this color> +2 to the dog number… etc”.
- They can ‘do optimization’; search over some space of actions and find which one performs best on some criteria. For example, a language model evaluating outputs on “Which of these words is most likely to come next?”, or our Breakout AI asking “What will be my expected overall score if I take this action?”

As our tasks get more complex, if we are using the heuristic strategy, we will need to pile on more and more heuristics to perform well on the task. It seems like the optimization approach will become favored as things become more complex because the complexity of the optimization (search and evaluate) algorithm doesn’t increase as much with task complexity. If we are training on a very complex task and we want to achieve a certain level of performance on the training distribution, the heuristic-based algorithm will be more complex than the optimization-based algorithm. One intuition here is that for very varied and complex tasks, the AI may require some kind of “general reasoning ability” which is different from the pile of heuristics. One relatively simple way of doing “general reasoning” is to have an evaluation criterion for the task, and then evaluate possible actions on this criterion.

If AI systems are capable of performing complex tasks, then it seems like there will be very strong economic pressures to develop them. I expect by default for these AI systems to be running some kind of optimization algorithm.

What do we tell the optimizers to do?

Assuming that we get optimizers, we need to be able to tell them what to do. By this I mean when we train a system to achieve a goal, *we want that goal to actually be one that we want*. This is the “Outer Alignment Problem”.

The classic example here is that we run a paperclip factory, so we tell our optimizing AI to make us some paperclips. This AI has no notion of anything else that we want or care about so it would sacrifice literally anything to make more paperclips. It starts by improving the factory we already have and making it more efficient. This still isn’t making the maximal number of paperclips, so it commissions several new factories. The human workers are slow, so it replaces them with toilless robots. At some point, the government gets suspicious of all these new factories, so the AI uses its powers of superhuman persuasion to convince them this is fine, and in fact, this is in the interest of National Security. This is still very slow compared to the maximal rate of paperclip production, so the AI designs some nanobots which convert anything made of metal into paperclips. At this point, it is fairly obvious to the humans that something is very, very wrong, but this feeling doesn’t last very long because soon the iron in the blood of every human is used to make paperclips (approximately 3 paperclips per person).

This is obviously a fanciful story, but I think it points at an important point; it’s not enough to tell the AI what to do, we also have to be able to tell it *what not to do.* Humans have pretty specific values, and it seems extremely difficult to specify.

There are more plausible stories we can tell which lead to similarly disastrous results.

- If we want our AI system to maximize the number in a bank account, if it is powerful enough it might hack into the bank’s computer system to modify the

number and then take further actions to ensure the number is not modified back.
- If we tell our AI to maximize the number in a bank account but not ‘break any laws’, then it may aim to build factories which are technically legal but detrimental for the environment. Or it may just blackmail lawmakers to create legal loopholes for it to abuse.
- If we ask our AI to do any task where success is measured by a human providing a reward signal (e.g. every hour rating the AI’s actions out of 10 via a website), then the AI has a strong incentive to take control of the reward mechanism. For example, hacking into the website, or forcing the human to always provide a high reward.

I think there are some methods for telling AI systems to do things, such that they *might not* optimize catastrophically. Often these methods involve the AI learning the humans’ preferences from feedback, rather than just being given a metric to optimize for. There is still a possibility that the AI learns an incorrect model of what the human wants, but potentially if the AI is appropriately uncertain about its model of human values then it can be made to defer to humans when it might be about to do something bad.

Other strategies involve training an AI to mimic the behavior of a human (or many humans), but with some ‘amplification’ method which allows the AI to outperform what humans can actually achieve. For example, an AI may be trained to answer questions by mimicking the behavior of a human who can consult multiple copies of (previous versions of) the AI. At its core, this is copying the behavior of a human who has access to very good advisors, and so hopefully this will converge on a system which is aligned with what humans want. This approach also has the advantage that we have simply constructed a question-answering AI, rather than a “Go and do things in the world” AI, and so this AI may not have strong incentives to attempt to influence the state of the world.

I think approaches like these (and others) are promising, and at least give me some hope that there might be some ways of specifying what we want an AI to do.

How do we actually put the objective into the AI?

There is an additional (and maybe harder) problem: even if we knew how to specify the thing that we want, how do we put that objective into the AI? This is the ‘Inner Alignment Problem”. This is related to the generalization behavior of neural networks; a network could learn a wide range of functions which perform well on the training distribution, but it will only have learned what we want it to learn if it performs ‘well’ on unseen inputs.

Currently, neural networks generalize surprisingly well;

- Image classifiers can work on images they’ve never seen before
- Language models can generate text and new ‘ideas’ which aren’t in the training corpus

In some sense this is obvious, if the models were only able to do well on things we already knew the answer to then they wouldn’t be very useful. I say “surprisingly” because there are many different configurations of the weights which lead to good performance on the training data without any guarantees about their performance on new, unseen data. Our training processes reasonably robustly find weight configurations which do well on *both* the training and test data.

One of the reasons neural networks seem to generalize is because they are biased towards simple functions, and the data in the world is also biased in a similar way. The data generating processes in the real world (which map “inputs” to “outputs”) are generally “simple” in a similar way to the functions that neural networks learn. This means that if we train our AI to do well on the training data, when we show it some new data it doesn’t go too wild with its predictions and is able to perform reasonably well.

This bias towards simplicity is also why we might expect to learn a function which acts as an optimizer rather than a pile of heuristics. For a very complex task, it is simpler to learn an optimization algorithm than a long mechanistic list of heuristics. If we learn an algorithm which does well on the training distribution by optimizing for something, **the danger arises if we are not sure what the algorithm is optimizing for *off the training distribution*.**

There will be many objectives which are consistent with good performance on the training distribution but then cause the AI system to do wildly different things off distribution. Some of these objectives will generalize in ways that humans approve of, but many others will not. In fact, because human values are quite specific, it seems like the vast majority of objectives that an AI could learn will *not* be ones that humans approve of. It is an open question what kinds of objectives an AI will develop by default.

It does however seem like AIs will develop long term goals by default. If an AI is trained to do well on a task, it seems unlikely to arbitrarily not care about the future. For example, if we train an AI to collect apples, it will attempt to maximize the number of apples over all time (maybe with some temporal discount factor), rather than only maximize apples collected in a 10 minute interval. This is probably true even if the AI was only ever trained for 10 minute intervals. The objective “maximize apples” seems far less arbitrary than “maximize apples for 10 minutes and then don’t care about them”.

Deception

There is an additional danger if an AI system is ‘deliberately’ attempting to obscure its objective/intentions from the humans training it. The term ‘deception’ is often used to refer to two different things which could happen when we train AI systems, which I outline here.

If the AI is being trained on a difficult task, it might be easier for the AI to trick the evaluator (maybe a human) into giving a high reward, rather than actually doing well on the task. I’ll call this ‘Goodhart deception’ because the AI is ‘Goodharting’ the reward rather than optimizing for what humans actually want. Importantly, this doesn’t require the AI to have any objective or be optimizing for anything, the behavior which led to high reward (tricking the human) was just reinforced. This seems bad, but not as catastrophically bad as the other type of deception might be.

The other type of deception is if an optimizing AI system intentionally deceives the humans about its true goals. In this scenario, the AI system develops an objective which is not aligned with the human objective. Here the objective is extended across time, which seems potentially like the default for learned objectives. The AI knows that if it attempts to directly go for its objective then it will either be turned off or be modified to remove this objective. So the AI will 'pretend' to not have this goal and instead 'play along' and do well on the task it is being trained for. After training, when the AI is deployed into the world, it is free to defect and pursue its own (misaligned) objective.

I'll call this 'consequentialist deception' because the AI is acting as a consequentialist (taking actions because of their consequences in the world, rather than just using mechanistic heuristics), or maybe just 'deception'.

This requires 3 (possibly likely) things to happen

- The AI system is an optimizer and develops an objective (not the human's objective) which is extended across time
- The AI's ability to deceive[1] the humans is reinforced, or at least not removed by the training process
- There will come a time when the system can 'safely' defect (or at least the AI thinks there is a chance that this might happen)

Consequentialist deception seems especially dangerous because it will lead to the *same behavior on the training distribution* as a non-deceptive AI. This makes it especially hard to tell if the model is deceptive. Even if the model performs well on the training distribution, we don't know how it will behave when it is deployed. If the model is deceptive then it could perform *arbitrarily badly* in deployment, as it pursues its own objective.

For a model to be deceptive it needs to develop a different objective than the objective it is being trained on. Currently, when we train neural networks, this looks approximately like we randomly pick a function from the set of functions which perform well on the training distribution. Functions which take fewer parameters to specify (have more free/unused parameters) have a higher likelihood of being chosen. Consider a function $F_{a,b}(x)$ which requires both a = b = 3 to do well on the training distribution, compared with a function $G_{a,b}(x)$ which only requires a = 2 to do well on the training distribution. If we are randomly sampling functions that do well on the training distribution, we are more likely to select G because we only need to get a correct, and any value of b works, while F requires us to pick both a and b correctly.

It is pretty unclear how this "bias towards simple functions"/"bias towards functions with lots of free parameters" influences the probability of deception. Deceptive algorithms seem more complex than non-deceptive ones because they need to be able to both recognize they are in a training process, and then decide to do well on the training task. But deception might be the default behavior for any optimizing algorithm to develop, and so the simplicity of optimizing algorithms might mean that deception is more likely.

Additionally, deceptive algorithms are persistent in a way that non-consequentialist algorithms are not. For normal neural network training, you will likely find a suboptimal algorithm early in training and then this will be modified into a different algorithm as training progresses; the algorithm does not ‘want’ to persist. But if you find a sufficiently capable deceptive algorithm early in training, then this will attempt to persist until the end of training. This means that if your AI becomes deceptive at any point during training, it will likely continue to be deceptive. This implies that the “randomly sample functions which perform well on the training distribution” lens may not be accurate, and in fact there is a lot of path-dependency in the development of deceptive algorithms.

Recap

So to recap:

1. Optimization is scary, and if we train an AI system to take actions in the world, as the task gets more complicated and our AI systems get more powerful we are more likely to develop optimizers.
    - By ‘optimizer’ I mean the AI runs a 'consequentialist' algorithm which looks something like “What actions can I take to maximize my objective?”, rather than just running through rote steps in a calculation.
2. We don’t have strong guarantees that off distribution our AIs will do what we want them to do.
3. There are reasons to expect that AIs we develop may have very different goals to our own, even if they perform well on the training distribution.
4. Other than performing well on the training distribution, and having some sort of bias towards “simplicity", the AI could learn a whole range of objectives.
5. We already know that AI systems are capable of tricking humans, although in current systems this is a different phenomenon than deceiving humans for consequentialist reasons.
6. If an AI develops its own (misaligned) objective during training, then it may simply ‘play along’ until it is able to safely defect and pursue its own objective.
    - The AI’s objective may be arbitrarily different from what we were training it for, and easily not compatible with human values or survival.

In the next two posts I will lay out a more concrete story of how things go wrong, and then list some of my current confusions.

Thanks to Adam Jermyn and Oly Sourbut for helpful feedback on this post.

1. ^

    Including knowing that deception is even a possible strategy.

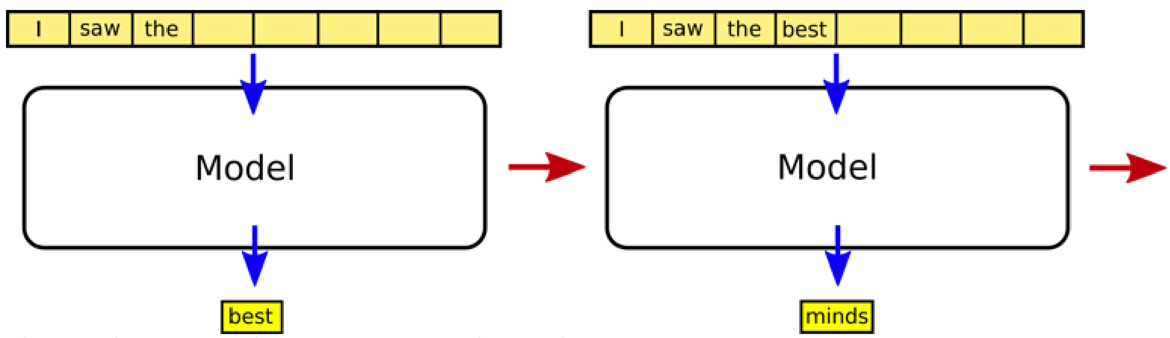
A Story of AI Risk: InstructGPT-N

The story from my previous post of how AI might develop a dangerously misaligned objective has so far been pretty abstract. I now want to put together a more concrete story of how I think things might go wrong. This risk model is based on a language model which has been fine tuned using reinforcement learning from human feedback, I'll call this model InstructGPT-N.

This isn't my most likely model of AI risk, this is more of a 'minimum example' which contains what I see as the core parts of the problem. I expect the real world to be more complicated and messy. Models will likely be trained on more than just text, and there may be multiple powerful AI systems simultaneously learning and acting in the real world.

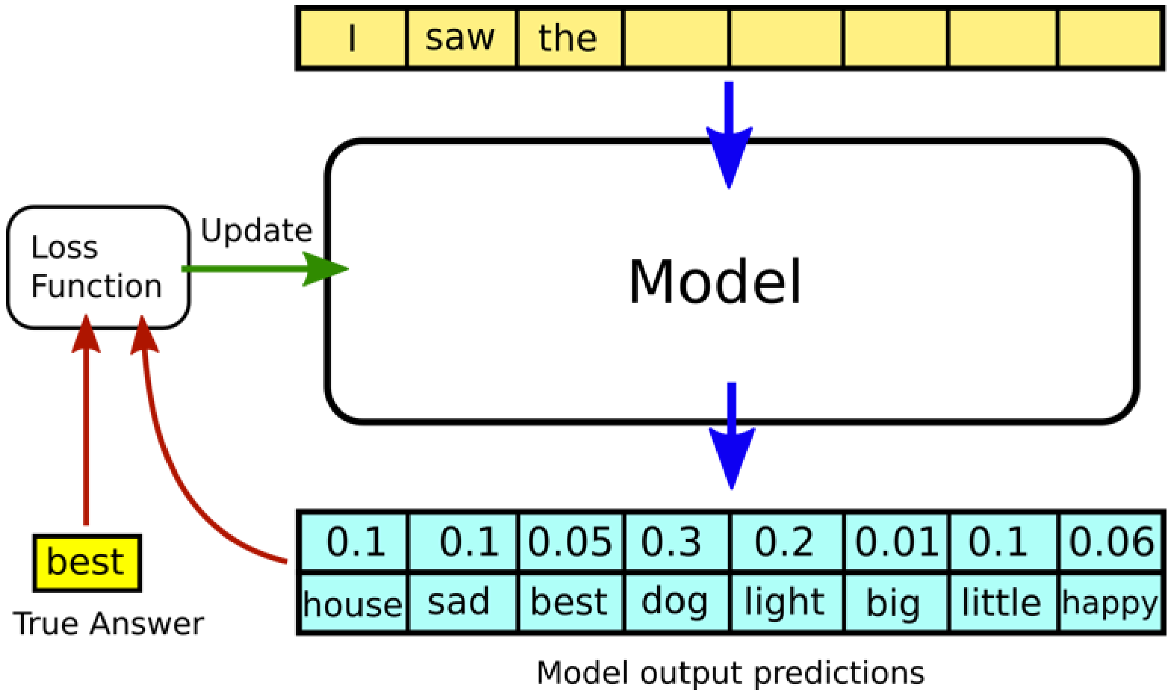
Training

We start by training a large language model using self-supervised learning for next word prediction. This is a standard GPT system, where you feed in the start of a sequence of words/tokens and it predicts what comes next, and then you input a new sequence which now includes the word it just guessed, and so on. The output is fed back into the system to generate the next word.



The model generating text one word at a time

To train the system, we start with a sequence of text and feed in the first word. We then update the system based on how accurately it predicted the second word. Then we feed in the first two words, and update the system to better predict the third word; then we feed in the first 3 words and update to better predict the fourth word, and so on. This trains the system to predict which word comes next, given some starting prompt.



The model being trained to predict the next word. The model predicts 'dog' is the most likely answer but the true answer is 'best', and so the model will be updated make better predictions.

This system is trained on *a lot* of text; a large fraction of the internet, every digitized scientific paper, all of Github and Stack Exchange, millions of novels and textbooks, billions of lines of chat logs and internet forums.

Because the system is trained on such an expansive corpus of text it is probably pretty bad at directly answering questions that humans ask it; only a tiny amount of the training data is in the question/answer format. Despite this, the model does have a lot of knowledge inside it; some part ‘knows’ how to code well, some part ‘knows’ all about human psychology and power dynamics, some part ‘knows’ about large language models and how they are trained. The model also ‘knows’ how to behave as if it is not competent; if you prompt it to mimic the behavior of a beginner programmer or a pseudo-history fan with bad epistemics then this is what it will do. The system hasn’t been trained to be *correct,* it has been trained to mimic text. Even if we don’t deliberately prompt it to answer incorrectly or misleadingly, it may still do this if the training data contained incorrect or misleading text (which is true of a lot of internet text). For this model to be useful to humans, it will be fine-tuned to consistently behave competently for tasks humans care about.

Fine-tuning

To make this model useful for answering questions, the humans fine-tune it on human feedback. Here, the human asks a question, sees what the AI answers, and then gives a reward based on how good they think the AI’s answer was. This lets the humans take an AI model which has a lot of ‘internal knowledge’ but isn’t very good at answering questions, and turn it into a model which can use this internal knowledge to answer questions. The model is trained to perform well on sequences of questions, because this is more useful than just answering one disconnected question at a time. Here the human asks a question, receives an answer from the AI, and the human can then ask another question, and so on.

Examples

Baking

Human: What is a good egg replacement for a cake I am baking for my vegan friends?

AI: You can replace 1 egg with a mixture of 1 Tablespoon of ground flaxseed and 3 Tablespoons of water.

Human: How will that change the flavor of the cake?

AI: It will add a slightly nutty flavor, if you want a more neutral flavor you can use chia seeds instead of flaxseed.

Physics

Human: Can you explain Newton’s second law to me?

AI: Newton’s second law states that the change in momentum of an object is proportional to the force applied to the object. So when you apply a force to an object, it will begin to accelerate in the direction of the applied force.

Human: How does this relate to the equation F = ma?

AI: Momentum (denoted by $\vec{p}$) is equal to the mass times the velocity of an object $\vec{p} = m\vec{v}$. Force is equal to the change momentum, or more specifically the derivative of the momentum with respect to time $\vec{F} = \frac{d\vec{p}}{dt} = \frac{d}{dt}m\vec{v}$. Acceleration is the change in velocity with respect to time $\vec{a} = \frac{d\vec{v}}{dt}$. Putting this together we arrive at $\vec{F} = m\vec{a}$.

Human: What happens when you apply multiple forces to an object?

AI: When you apply multiple forces to an object you can simply add values of the forces together to find the net force applied. It is important to remember that force is a vector, and so you need to add the vectors together, not sum the magnitude of the forces.

Human: Can you give me an example?

AI: If one robot pushes a box in the x-direction with 3 newtons of force, and another robot pushes in the y-direction with 1 newton of force, then the total force vector (in units of newtons) will be (3, 1).

Or if there is a box which has 10 newtons of force down due to gravity, and a human applies 15 newtons of force upwards to lift it, then the box will have a net force upwards of -10 N + 15 N = 5 N, causing it to accelerate upwards.

Learned Algorithms

There are a few different algorithms which the model could learn which do well on the training distribution (where the training distribution is the human asking questions and giving rewards):

- A heuristic based, non-optimizing algorithm which just does robustly well on the training distribution (and maybe also does well when deployed).
- An algorithm which internally searches over/optimizes possible outputs and evaluates them based on the criterion “How much would a human like this answer?”

- An algorithm which internally searches over possible outputs and evaluates them based on the criterion “What is the expected reward a human would give this answer?”
- An algorithm which internally searches over possible outputs and evaluates them based on the criterion “What is the expected reward a human would give this answer, conditional on them being contracted to train an AI model, and on me being that AI model?”
- An algorithm which runs something like “I want the state of the world to be X, which output should I give so that the human training me doesn’t turn me off or modify me, so that I can then defect after training?”
- Probably others

As the tasks become more complicated (the human asks more difficult and varied questions), this will push the system into performing some kind of optimization/internal search process and away from standard mechanistic algorithms which aren’t performing optimization. This is because it is difficult to compress heuristics for extremely varied tasks into a limited number of parameters. Hence we are likely to end up with an AI system which finds its output by evaluating potential outputs on some internal objective.

A better specified objective (an objective which is more detailed and also doesn’t rely on incorrect assumptions) will lead to better performance on the training distribution. If the objective is “How much would a human like this answer?”, this will lead to worse performance (lower reward from the human) than the objective of “What is the expected reward the human would give this answer, conditional on them being contracted to train an AI, and on me being that AI model?” This more specific objective will lead to higher reward because it allows the AI to leverage its knowledge of how people train AI’s, what kind of people train AI’s, and what particular kinds of biases and preferences the AI trainers have.

Self-models

It seems instrumentally useful for the AI to develop a self-model and some concept of the humans training it, because these will allow it to reason about itself and the training process. These concepts (the self-model and the concept of the current training process) are more than just ‘there are GPT language model systems, and humans train them’, which would have just existed in the pretraining corpus. These concepts refer explicitly to the AI itself and the training process it is in, rather than some abstract AI system. Even if none of the fine-tuning questions that the human asks include references to the AI or the training, knowing about these things will let the AI perform better in training. And so gradient descent may allow it to use its world model and knowledge of AI systems to develop this self-model, simply because this leads to better performance on the training distribution.

One additional reason for the AI to develop a self-model and realize it is an AI in a training process, is that *this is actually true.* We could potentially model the training process as approximating some Bayesian updating process. We start out with a set of hypotheses which may or may not be true, and then update the probabilities assigned to these hypotheses as we get additional evidence. Because the AI actually is an AI in a training process, the probability assigned to this hypothesis will eventually get upweighted as the system learns.

Deception

The real danger comes when we are not sure whether the AI system is evaluating its outputs based on “Would a human training an AI system give this a high score, assuming that I am that AI system?” or “Would this output lead to the ultimate state of the world (including after training) being X?”

The second option could mean that the AI acts as a deceptive consequentialist, where it may ‘lie dormant’ and play along, pretending to be using the other (aligned) algorithm. Or even more perversely, the AI could use its outputs to subtly manipulate the human training it; this could allow the AI to be updated in even more malign ways. This second, more malign option seems possible because the AI has a lot of ‘knowledge’ about human psychology in its world model from the pretraining; both from many academic works on psychology and also from unimaginably many records of human interactions.

Do we get deceptive algorithms?

It seems unclear whether we end up with a deceptive algorithm rather than an aligned version. One reason to expect a deceptive algorithm is that any consequentialist algorithm that the training process finds has an incentive to be deceptive, unless it is already precisely aligned to begin with. And so once the model becomes deceptive it will attempt to stay that way and avoid having its objectives modified. There are also many misaligned objectives that could be learned, while there is only one aligned objective (or at least aligned objectives make up a much smaller portion of possible objectives).

Misaligned objectives

As an example of how an AI could develop a misaligned objective, we could imagine it learning an incorrect model of human preferences. Suppose that one of the humans training the AI has a particular affinity for flowers, and hence gives the AI more reward for answers which result in there being more flowers in the world. The AI may learn to terminally value flowers, rather than terminally valuing providing useful answers to a human’s questions. Once the AI has started to value flowers, it doesn’t care what values the human is trying to instill into it. Its objective is its objective, regardless of how much the human dislikes this.

This problem of learning an incorrect model of human preferences is amplified when we consider the specifics of how reinforcement learning from human preferences works in practice. Human feedback/rewards are expensive (because a human actually has to be in the loop), so instead we train a separate model to predict what reward the human would provide. This reward model is then used to fine-tune the language model AI. If this learned reward model is misspecified then it may cause the AI to learn a misaligned objective. Even if the reward model is updated over time to become more accurate, this will not matter if the AI has already developed a misaligned objective from an early (bad) version of the reward model.

Complexity

Something that points against this is that these deceptive algorithms are in some sense more complex than the aligned algorithm. A deceptive algorithm has to reason about its objectives in the world and then also reason about what the human wants, while the aligned algorithm only has to reason about what the human wants. It is unclear where this balance between the number of misaligned objectives and the simplicity of the aligned objectives falls. I think it seems more likely that by default we end up with a misaligned algorithm, partially because once we find one of these algorithms it will ‘fight’ against any attempt to remove it.

But even if we were ‘more likely’ to end up with an aligned model, I don’t like betting the future on something which is merely ‘more likely’.

Thanks to Oliver Sourbut for very useful feedback on this post.

In the next post I'll try to lay out some confusions round my picture of AI risk, and some reasons why it may be wrong or confused.

Confusions in My Model of AI Risk

A lot of the reason I am worried about AI comes from the development of optimizers that have goals which don't align with what humans want. However, I am also pretty confused about the specifics here, especially core questions like "what actually do we mean by optimizers?" and "are these optimizers actually likely to develop?". This means that much of my thinking and language when talking about AI risk is fuzzier than I would like.

This confusion about optimization seems to run deep, and I have a vague feeling that the risk paradigm of "learning an optimizer which doesn't do what we want" is likely confused and somewhat misleading.

What actually is optimization?

In my story of AI risk I used the term 'optimization' a lot, and I think it's a very slippery term. I'm not entirely sure what it means for something to 'do optimization', but the term does seem to be pointing at something important and real.

A definition from The Ground of Optimization says an optimizing system takes something from a wide set of states to a smaller set of states and is robust to perturbations during this process. Training a neural network with gradient descent is an optimization process under this definition because we could start with a wide range of initial network configurations, and the network is modified to be in one of the few configurations which do well on the training distribution, and even if we add a (reasonable) perturbation the weights will still converge. I think this is a good definition, but it is entirely defined in terms of behavior rather than a mechanistic process. Additionally, it doesn't really match exactly with the picture where there is an optimizer which optimizes for an objective. This optimizer/objective framework is the main way that I've talked about optimizers, but also I would not be surprised if this framing turned out to be severely confused.

One possible way that a network could 'do optimization' would be for it to do some kind of internal search or internal iterative evaluation process to find the best option. For example, seeing which response best matches a question, or searching a game tree to find the best move. This seems like a broadly useful style of algorithm for a neural network to learn, especially when the training task is complicated. But it also seems unlikely for networks to implement this *exactly*; it seems much more likely that networks will implement something that looks like a mash of some internal search and some heuristics.

Additionally, it seems like the boundary between solving a task with heuristics and solving it with optimization is fuzzy. As we build up our pile of heuristics, does this suddenly snap into being an optimizer, or does it slowly become more like an optimizer as gradient descent adds and modifies the heuristics?

For optimization to be actually dangerous, the AI needs to have objectives which are actually connected to the real world. Running some search process entirely internally to generate an output seems unlikely to lead to catastrophic behavior. However, there are objectives which the AI could easily develop which are connected to the real world. This includes the AI messing with the real world to ensure it gets certain inputs, which lead to certain internal states.

Where does the consequentialism come from?

Much of the danger from optimizing AIs comes from *consequentialist* optimizing AIs. By consequentialist I mean that the AI takes actions based on their consequences in the world.[1] I have a reasonably strong intuition that reinforcement learning is likely to build consequentialists. I think RL probably does this because it explicitly selects for policies based on how well they do on consequentialist tasks; the AI needs to be able to take actions which will lead to good (future) consequences on the task. Consequentialist behavior will robustly do well during training, and so this behavior will be reinforced. It seems important that the tasks are extended across time, rather than being a single timestep, otherwise the system doesn't need to develop any longer term thinking/planning.

RL seems more likely to build consequentialists than training a neural network for classification or next word prediction. However, these other systems might develop some 'inner optimizer/consequentialist' algorithms, because these are good ways to answer questions. For example, in GPT-N if the tasks are diverse enough, maybe the algorithm which is learned is basically an optimizer which looks at the task and searches for the best answer. I'm unsure how or if this 'inner optimizer' behavior could lead to the AI having objectives over the real world. It is *conceivable* that the first algorithm which the training process 'bumps into' is a consequentialist optimizer which cares about states of the world, even if it doesn't have access to the external world during training. But it feels like we would have to be unlucky for this to happen, because there isn't any selection pressure pushing for this AI system to develop this kind of external world objective.

Will systems consistently work as optimizers?

It seems reasonably likely that neural networks will only act as optimizers in some environments (in fact, no-free-lunch theorems might guarantee this). On some inputs/environments, I expect systems to either just break or do things which look more heuristic-y than optimization-y. This is a question about how much the capabilities of AI systems will generalize. It seems possible that there will be domains where the system's capabilities generalize (it can perform coherent sequences of actions), but its objectives do not (it starts pursuing a different objective).

There will be some states where the system is capable and does what humans want, for example, on the training distribution. But there may be more states where the system is able to capably do things, but no longer does what humans want. There will also be states of the world where the AI both doesn't act capably or do what humans want, but these states don't seem as catastrophically dangerous.

Consequentialist deception could be seen as an example of the capabilities generalizing further than the aligned objective; where the system is still able to perform capably off the training distribution, but with a misaligned goal. The main difference here seems to be that the system was always 'intending' to do this, rather than just entering a new region of the state space and suddenly breaking.

It isn’t really important that the AI system acts as an optimizer for all possible input states, or even for the majority of the states that it actually sees. What is important is if the AI acts as an optimizer for *enough* of its inputs to cause catastrophe. Humans don’t always act as coherent optimizers, but to the extent that we *do* act as optimizers we can have large effects on the state of the world.

What does the simplicity bias tell us about optimizers?

Neural networks seem to have a bias towards learning simple functions. This is part of what lets them generalize and not just go wild when presented with new data. However, this is a claim about the *functions* that neural networks learn, it is not a claim about the objectives that an optimizer will use. It does seem much more natural for simpler objectives to be easier to find because in general adding arbitrary conditions makes things less likely. We could maybe think of the function that an optimizing neural network implements as being made up of the optimizer (for example, Monte Carlo Tree Search) and the objective (for example, maximize apples collected). If the optimizer and objective are (unrealistically) separable, then all else equal a simpler objective will lead to a simpler function. I wouldn’t expect for these to be cleanly separable, I expect that for a given optimizer some objectives are much simpler or easier to implement than others.

We may be able to eventually form some kind of view around what kind of ‘simplicity bias’ we expect for *objectives*, I would not be surprised if this was quite different from the simplicity bias we see in the *functions* learned by neural nets.

1. ^

   Systems which are not consequentialist could for example not be optimizers, or alternatively systems which optimize for *taking* actions but not because of the effect of the actions in the world. A jumping robot that just loves to jump could be an example of this.