Security amplification

Paul Christiano Follow
Oct 26, 2016 · 15 min read

An apparently aligned AI system may nevertheless behave badly with small probability or on rare "bad" inputs. The reliability amplification problem is to reduce the failure probability of an aligned AI. The analogous **security amplification problem** is to reduce the prevalence of bad inputs on which the failure probability is unacceptably high.

We could measure the prevalence of bad inputs by looking at the probability that a random input is bad, but I think it is more meaningful to look at the *difficulty of finding a bad input*. If it is exponentially difficult to find a bad input, then in practice we won't encounter any.

If we could transform a policy in a way that multiplicatively increase the difficulty of finding a bad input, then by interleaving that process with a distillation step like imitation or RL we could potentially train policies which are as secure as the learning algorithms themselves—eliminating any vulnerabilities introduced by the starting policy.

For sophisticated AI systems, I currently believe that meta-execution is a plausible approach to security amplification. (ETA: I still think that this basic approach to security amplification is plausible, but it's now clear that meta-execution on its own can't work.)

Motivation

There are many inputs on which any particular implementation of "human judgment" will behave surprisingly badly, whether because of trickery, threats, bugs in the UI used to elicit the judgment, snow-crash-style weirdness, or whatever else. (The experience of computer security suggests that complicated systems typically have *many* vulnerabilities, both on the human side and the machine side.) If we aggressively optimize something to earn high approval from a human, it seems likely that we will zoom in on the unreasonable part of the space and get an unintended result.

What's worse, this flaw seems to be inherited by any agent trained to imitate human behavior or optimize human approval. For example,

inputs which cause humans to behave badly would also cause a competent human-imitator to behave badly.

The point of security amplification is to remove these human-generated vulnerabilities. We can start with a human, use them to train a learning system (that inherits the human vulnerabilities), use security amplification to reduce these vulnerabilities, use the result to train a new learning system (that inherits the reduced set of vulnerabilities), apply security amplification to reduce those vulnerabilities further, and so on. The agents do not necessarily get more powerful over the course of this process—we are just winnowing away the idiosyncratic human vulnerabilities.

This is important, if possible, because it (1) lets us train more secure systems, which is good in itself, and (2) allows us to use weak aligned agents as reward functions for a extensive search. I think that for now this is one of the most plausible paths to capturing the benefits of extensive search without compromising alignment.

Security amplification would not be directly usable as a substitute for informed oversight, or to protect an overseer from the agent it is training, because informed oversight is needed for the distillation step which allows us to iterate security amplification without exponentially increasing costs.

Note that security amplification + distillation will only remove the vulnerabilities that came from the human. We will still be left with vulnerabilities introduced by our learning process, and with any inherent limits on our model's ability to represent/learn a secure policy. So we'll have to deal with those problems separately.

Towards a definition

The security amplification problem is to take as given an implementation of a policy **A**, and to use it (along with whatever other tools are available) to implement a significantly more secure policy $\mathbf{A}^+$.

Some clarifications:

- "implement:" This has the same meaning as in capability amplification or reliability amplification. We are given an implementation of **A** that runs in a second, and we have to implement $\mathbf{A}^+$ over the course of a day.

- "secure": We can measure the security of a policy **A** as the difficulty of finding an input on which **A** behaves badly. "Behaves badly" is slippery and in reality we may want to use a domain-specific definition, but intuitively it means something like "fails to do even roughly what we want."

- "more secure:" Given that difficulty (and hence security) is not a scalar, "more secure" is ambiguous in the same way that "more capable" is ambiguous. In the case of capability amplification, we need to show that we could amplify capability in *every* direction. Here we just need to show that there is *some* notion of difficulty which is significantly increased by capability amplification.

- "significantly more secure": We would like to reach very high degrees of security after a realistic number of steps. This requires an exponential increase in difficulty, i.e. for each step to multiplicatively increase the difficulty of an attack. This is a bit subtle given that difficulty isn't a scalar, but intuitively it should take "twice as long" to attack an amplified system, rather than taking a constant additional amount of work.

- Security amplification is probably only possible when the initial system is sufficiently secure—if random inputs cause the system to fail with significant probability, then we are likely to be out of luck. This is analogous to reliability amplification, which is only possible when initial system is sufficiently reliable. Under the intended interpretation of "security," humans are relatively secure; we can implement a policy **H** which is relatively hard to exploit (e.g. which humans aren't capable of reliably exploiting). So humans suffice to get the ball rolling.

Capability amplification vs. security amplification

If we interpret "capability" broadly, then capability amplification subsumes security amplification. Moreover, I expect the two problems to be solved by the same mechanism (unlike reliability amplification, which probably requires something completely different). So in some sense it is most natural to think of capability and security amplification as a single problem.

But I think that security amplification has different implications, may require a different style of analysis, and may be possible or impossible independently of other parts of capability amplification. And in general I think it is good practice to try to split up a strong claim into several weaker claims, even if after establishing each of the

weaker claims you will just have to prove a completely new theorem that generalizes all of them. The weaker claims give us a nice testing ground in which to find and resolve some of the difficulties before having to confront the whole problem.

Security amplification by meta-execution

I am hopeful that security can be amplified by something like meta-execution.

In meta-execution, the meta-executor never directly looks at the whole system's input. Instead, it looks at small parts of the input in isolation, and decides for itself how to build those up into a representation of the input.

Similarly, the meta-executor never directly experiences the entire reasoning process; the actual state of the reasoning process is maintained by a large number of meta-executors working in parallel, and each one is responsible for implementing a small part of the reasoning process.

With the exception of the meta-executors who directly observe small pieces of the input, the actual inputs provided to a meta-executor are the product of two forces:

1. The input to the overall system.
2. The optimization applied by the meta-executor as it process that input. The goal of the meta-executor is to ensure that it will be able to correctly handle everything that it sees during the meta-execution, so it is trying to avoid creating states that would exploit vulnerabilities of the meta-executor.

The hope is that an adversary who controls [1] and wants to cause the meta-executor to behave badly must do additional work in order to overcome factor [2].

Realistically, I think that the largest question about this procedure is whether meta-execution can work at all while preserving alignment.If meta-execution can work at all, then I think that there is a reasonable chance that it can also improve security.

That said, even if meta-execution works, it will probably be pretty hard to tell whether it actually reduces security.

As with capability amplification, I think that our goal should be to either convince ourselves that security amplification works, or else to identify an **obstruction**: a type of vulnerability that doesn't become significantly harder to exploit after a step of security amplification.

The rest of this post will try to explain why I think it is plausible that meta-execution could make vulnerabilities more difficult to exploit.

Examples

It seems helpful to consider a bunch of particular examples of vulnerabilities and think about how they behave under meta-execution.

I'll discuss a few examples here: a "magic phrase" which cause the agent **A** to fail badly when encountered, and an "unreasonably compelling argument" which **A** finds compelling even though it should not, and an "appeal to a broken intuition" which exploits a single step of **A**'s reasoning that cannot be broken down further or abstracted. I won't be able to argue convincingly that meta-execution could address any of these hypothetical vulnerabilities. But hopefully I can explain why it seems plausible to me that it would.

(*Warning: the rest of the post is somewhat informal and rambling*.)

Example: magic phrase

Setup. Suppose that the agent **A** fails whenever it sees the phrase "not playing solitaire."

If we had time to do enough capability amplification, then the amplified agent may eventually be able to discover this vulnerability itself and explicitly counteract it. This is not valid security amplification though; it does not help us amplify the security of a low-capability agent **A** in a reasonable amount of time. To amplify security we need to be able to fix vulnerabilities without discovering what they are (both because we occupy the position of the interior, and because we want to be secure against adversaries smarter than we are).

To make the problem harder we could assume that **A** actually fails whenever it merely *thinks of* the phrase "not playing solitaire," and that there is no advance warning or ability to see what is coming. For example, if the letters were revealed one at a time to **A**, then nothing might happen until "not playing so…" is revealed, at which point the

agent might imagine several possible completions, chance across "not playing solitaire," and then immediately fail.

Cause for optimism. If a string including "not playing solitaire" is given to the meta-executed system, then **A** won't directly observe it. Instead **A** must process a message like "the string with list of characters [x]." **A** could parse this string into words, obtaining a message like "the sentence with list of words [x]," where each word is represented by a copy of **A** who has seen that word and can answer questions about it. Then **A** could parse this sentence to form a representation its meaning, e.g. as "the assertion that [x] satisfies property [y]," with y pointing to "does not satisfy the property [z]", z pointing to "is currently performing action [w]" and so on.

In this way, it seems quite plausible that a sentence could be "understood" without any individual agent actually having to look at or even imagine the sequence of letters it contains.

Of course we do have to look at a whole bunch of words in the process of parsing a sentence, and we will have to look at many more if we want to actually do anything with it. So why think the situation is any better?

By hypothesis the agent doesn't know about the magic words and so can't actively avoid them; but the agent *can* treat its input with skepticism and try to prevent an adversary from having undue influence over the exact sentences that it produces. To that end, the agent is free to behave stochastically.

If successful, randomization might make it hard to force the agent to say *any particular* really unusual phrase with significant probability. (If the magic words aren't an unusual phrase, then it seems fair for us to give up, based on the initial agent not clearing the required initial level of security. E.g. if the agent fails whenever it sees or thinks the phrase "one more," we *probably* can't really help it.)

Hopefully meta-execution has shifted the attacker's problem from "say the magic phrase" to "engineer a situation which will cause the meta-executor to say the magic phrase with relatively high probability despite its efforts to behave randomly." It seems likely that this latter problem is quite a bit harder, suggesting that a magic phrase is not a plausible obstruction in the sense described above.

Other remarks. Of course, over the course of a larger computation it becomes more likely that a magic phrase will appear at one point or

another, even if the attacker isn't able to engineer a decisive attack. But as long as the magic phrase is not actually *likely* to appear, then this probability can hopefully be driven back towards zero with reliability amplification. We only need to prevent attacks that succeed with significant probability over the agent's stochasticity.

This process may sound unreasonably expensive. For example, if parsing a thousand word message, would we need to break it down into individual words and parse them one by one extremely carefully?

The point of *iterating* security amplification is that we don't have to do this all in one go. When parsing a thousand word message we can break it down into sentences and feed whole sentences to the agent. When processing sentences we can break them down into words. In general, the smaller the task the more careful we can be.

Example: unreasonably compelling arguments

Setup. Suppose that we want to build a system for evaluating arguments about what to do. We think that our starting agent **A** is pretty good—if it sees an argument made by a human, we expect its beliefs to reliably get more accurate rather than less accurate.

But now we would like to evaluate arguments made by more powerful intelligences than humans. We are concerned that some of these arguments may be *unreasonably compelling*—they would cause us to be convinced without being normatively informative. As an extreme case, there may be unreasonably compelling arguments that could convince us to adopt essentially arbitrary positions.

We cannot be protected from an unreasonably compelling argument by changing the way we handle individual words. At some point we are actually going to have to actually evaluate the argument, and that is where we'll run into trouble.

Cause for optimism. In order for meta-execution to handle this problem, we would need to be able to meta-execute the actual evaluation of the argument. For example, rather than inspecting a claimed syllogism and consulting our intuition to determine whether it seems valid, we would need to decide abstractly how to process a question like "does conclusion [x] follow from premises [a] and [b]?" where all of x, a, and b are messages representing parts of the argument.

Of course we could evaluate a proposed syllogism by simply unpacking all of its parts and consulting our intuition to determine whether it seems valid. The first question is: can we can do anything more abstract, that doesn't require looking directly at the whole input? The second question is: if we evaluate an argument in a more abstract way, are we actually more secure?

With respect to the first question: In general I believe that we *can* come up with at-least-slightly abstract procedures for evaluating arguments, which we believe are more accurate than a direct appeal to our intuitions. Although it would obviously be nice to have some convincing theoretical account of the situation, it looks like a largely empirical question. Fortunately, it's an empirical question that can be answered in the short term rather than requiring us to wait until powerful AI systems are available.

With respect to the second question: I think the key property of "unreasonably convincing" arguments is the following. Suppose that you tell me that I will hear an argument from source S, that I will evaluate it *correctly* (knowing that it came from source S), and that I will then come to believe X. After hearing this, I will simply accept X. An evaluation of an argument seems *incorrect* if, given a full understanding of the evaluation process, I wouldn't think that I should have been persuaded.

Now suppose that I find some argument convincing. And suppose that after lightly abstracting my evaluation process it still seems convincing—that is, I look at a sequence of steps like "I concluded that [x] followed from [a] and [b]." and I feel like, in light of that sequence of steps, I was correct to be convinced. It seems to me that then one of two things could be going wrong:

- One of these individual steps was wrong—that is, I asked "Does [x] follow from [a] and [b]?" and got back the answer "It sure does," but only because this step had unreasonably convincing aspects inside of it. It seems like this problem can be fixed by further secure amplification operating on the reasoning with a single step. (Just like we previously discussed breaking a paragraph into sentences, and then making the handling of sentences more secure by breaking sentences down into words.)
- I was incorrectly evaluating the abstract argument—I was misled about whether that sequence of steps *should have been* convincing.

I think the second category is most interesting, because it suggests the possibility of a kind of fixed point. An attacker could construct an argument which convinces me, and such that when I look at an abstracted version of my evaluation process I think that I ought to have been convinced, and when I look at an abstracted version of *that* evaluation process, I think that it *also* was convincing, and so on down the line.

If there is really such a fixed point, then that would be an obstruction to security amplification. After any number of iterations of security amplification, the difficulty of finding an attack would still be upper bounded by the difficulty of finding this fixed point.

I am actually not quite sure what we should make of such a fixed point. Certainly it is conceivable that there could be a plainly wrong argument which had this character. But I have a really hard time constructing a plausible example, and this fixed point does quite strongly suggests that an argument is "convincing for endorsed reasons."

(As mentioned earlier, it seems much more likely that we simply won't be able to evaluate arguments using meta-execution, and will be reduced to evaluating them directly.)

If an argument continues to be convincing in light of careful meta-execution, I am not totally sure whether we should call it "unreasonably" convincing. For now it seems like we need a better understanding of the situation, but it is certainly not obvious that there is a problem.

Example: appeal to a broken intuition

Setup. When evaluating an argument or making a decision, we sometimes seem to make a "bare" appeal to intuition—we pose a question to ourselves and discover our intuitive response, without any ability to break the process down further our scrutinize the source of the intuition.

Meta-execution is only feasible if sufficiently "big" intuitions can reliably be broken down into smaller pieces. But nevertheless this process may bottom out at some scale, e.g. perhaps there are questions that can be posed to a human in twenty seconds, where the human has an intuitive answer, but can't provide any additional detail.

It may be that for the most part these intuitive responses are correct, but that some of them are extremely unreliable. For example, in the process of considering a thought experiment about simulations and torture a person may freak out and give an irrational answer.

If we want to get the benefits from consulting human intuition, then it seems like an adversary could set up a situation where the most important intuitive queries exploit some vulnerability.

Cause for optimism. Even if the internal structure of an intuitive reaction is opaque, there are a number of ways that the surrounding argument can be changed in order to improve security:

- We can query the agent on random variants of an input, with many details varied, in order to avoid problems specific to a particular formulation. In some sense we are left only with cases where the core intuition is itself a vulnerability.
- We can inspect the input (and its meaning) abstractly, to search for any reasons why this particular appeal to intuition may be unreliable.
- We can treat the agent's reaction as just one piece of evidence that bears on a case. If we learn there is a thought experiment such that considering it causes the agent to conclude "doing X is extraordinarily important" then that constitutes compelling but not overwhelming evidence.
- We can consider the agent's response to a range of similar inputs, and make inferences from those responses in the aggregate.

If all of these tests are insufficient to protect us from the purported vulnerability, then again I think it's not really clear on what grounds we are describing it as a "vulnerability." It's not obvious that we should rest comfortably at this point, but at least it seems plausible that we should.

Empirical tests

The discussion in the last section was very vague and intuitive, but fortunately the actual claims at issue seem to empirically accessible. It is very easy to implement meta-execution using humans as the meta-executor. As a result:

- We can *just test* whether we can evaluate arguments or make decisions abstractly in a way that seems at least as good, and preferably better, than evaluating them directly.
- We actually pick a simple idea, and see whether a human meta-executor can abstractly make decisions without ever encountering that idea (even on adversarial inputs).

Mostly I think that many of these issues will become quite obvious as we get some practical experience with meta-execution (and hopefully it will also become clear how to get a better theoretical handle on it).

Last summer I actually spent a while experimenting with meta-execution as part of a metaprogramming project dwimmer. Overall the experience makes me significantly more optimistic about the kinds of claims in the post, though I ended up ambivalent about whether it was a practical way to automate programming in the short term. (I still think it's pretty plausible, and one of the more promising AI projects I've seen, but that it definitely won't be easy.)

Conclusion

We can attempt to quantify the security of a policy by asking "how hard is it to find an input on which this policy behaves badly?" We can then seek security amplification procedures which make it harder to attack a policy.

I propose meta-execution as a security amplification protocol. I think that the single biggest uncertainty is whether meta-execution can work at all, which is currently an open question.

Even if meta-execution *does* work, it seems pretty hard to figure out whether it actually amplifies security. I sketched a few types of vulnerability and tried to explain why I think that meta-execution might help address these vulnerabilities, but there is clearly a lot of thinking left to do.

If security amplification could work, I think it significantly expands the space of feasible control strategies, offers a particularly attractive approach to running a massive search without compromising alignment, and makes it much more plausible that we can achieve acceptable robustness to adversarial behavior in general.