Designing a protocol to constraint AGI freedom

Cristian Curaba

February 18, 2024

Abstract

In this article (still a work in progress, for now, it's just an idea), I will propose to reason about AGI by considering freedom as a fundamental key concept. By considering freedom as a characteristic of an AGI, we may find new ways of controlling it and handling risks.

I will also propose a way to constrain the AGI system and handle risky behaviours for humans. The key idea is to force the AGI system to send outputs through a programmed protocol. This may be done through a penalization in the goal function during the learning phase (or fine-tuning it at the end)[1]. We can avoid adversarial or unauthorized protocol updates by penalizing them in the same way.

The protocol can implement a monitoring procedure with automatic "anomaly" detection which will warn the programmers[2]. It's also a way to allow communication between programmers and the AGI model, allowing correcting feedback. The protocol can be updated safely (avoiding adversarial updates) thanks to the digital signature of the programmers. Therefore, we can solve the stop button problem by updating the protocol.

Criticals are discussed at the end.

1 Why freedom?

In the AGI literature, two key, orthogonal concepts are popular to describe an AGI system: intelligence and goal. The intelligence is defined as the ability to achieve a goal. The goal is concretely formalized by the score/loss/cost/goal function which is one that any model tries to optimize[3]. Wide and general goals characterize the AGI systems. Literature is trying to align the AGI systems with human desire but there are still no clear solutions.

I suggest to consider a third key concept: freedom. Freedom is the variety of actions that the system can perform in the world to achieve its goal. The power of the AGI system may be limited by the freedom that we (the company

[1] It's not straightforward how to do it since it is dependent on the learning technique applied. Moreover, we want to structure the goal function in a way that any task decomposition without going through the protocol is disentivized.

[2] This type of prediction is challenging and, when the AGI is misaligned, it will try to deceive and hide behaviours we consider risky.

[3] This is only partially true: see mesa-alignment (or inner alignment) problem.

programmers) have granted to it[4]. In the state of the art of foundation models, even if they seem to partially understand the world and show some intelligent behaviours, their freedom is still forcefully constrained: they can just answer input questions. I think this is the main reason why actual models, for most people, are not considered dangerous for humans: they are not free to act in the world and the space of action is extremely limited.

Can we design a way to constrain future AGI systems to be safe for humans? We, of course, look for constraints that companies are incentivized to adopt: killing the AGI possibility to act in the world is not a useful option. Bounding resources (still no clear way to do it), like computational power, is not a solution: it's a matter of time before another company will train an AGI model without these constraints (unless AI governance goes really well). A possible branch of research is trying to train an AGI which, by design, avoids "extreme" best decisions (in high-dimensional space, if you hardly look for The Best Solution you will probably end up with some awkward and dangerous one) and find a constraining technique (similar to a regularization term but applied after the learning phase) to avoid them (look for quantilization).

One desired "toy" propriety is the shutdownability (also known as the "stop button problem" [Soa+15]). We also want to effectively constrain copies or submodules of the AGI system.

In the following section, I will design a way to constrain the AGI system and, as by product, to solve the stop button problem.

2 Building a protocol with digital signature

The key idea is to force the AGI system to send outputs through a protocol whenever wants to interact with external devices. The protocol can be updated safely by the company programmers based on the digital signature of the programmers. Check digital signature schemes [AZJ06] for more details. To achieve this, I am supposing that the AGI system is equipped with a local and secure memory where the public keys of the company programmers and the whole protocol are stored[5]. Such a protocol may also include a monitoring system to evaluate the AGI outputs.

To ensure that the AGI system will go through the protocol, we will need to train it to do so by penalizing (virtually $-\infty$) external interactions that are not sent through the protocol. We also need to penalize any attempt to modify the protocol without the digital signature of the company programmers. In this way, the AGI system will learn to send outputs through the protocol "by design" and it will avoid any unauthorized modifications. Adversarial attacks on the protocol will be ineffective because the AGI system will update the local

[4] Notice that the AGI space of action, at first approximation, is limited by nature law (e.g. it needs hardware to function) and design bottlenecks (e.g. constrained in a local network, before the deployment.)

[5] We must make sure, when designing the goal function for this purpose, to uniquely identify the correct address where the original protocol is stored.

protocol only if the digital signature of the company programmers is correctly applied. We will need to carefully design the goal function to disentivize the creation of AGI submodels (or copies) which won't send outputs through the protocol [6]. To clarify, we want to avoid the AGI splitting in submodules which interacts with the world without sending outputs through the protocol: the goal function cannot be designed simply as a sum of different "independent" purposes[7].

3 Solving the stop button problem

How can we design a protocol to solve the stop button problem? We (the company programmers) can just update the protocol to encrypt the AGI outputs in such a way that only the company programmers can decrypt them. In this way, instead of shutting down the AGI system "physically", we effectively design a way to make its outputs completely useless (basically a noise) while being able to communicate with the programmers to re-align it (if possible).

4 Other application of the protocol

The protocol can be used to monitor the AGI system. For example, we can design a protocol to monitor the AGI system and send a warning to the programmers if the AGI system is going to do something dangerous.

We must keep in mind that the AGI can bypass the protocol by delegating communications to external devices which have already established communication with the AGI (fundamentally as a VPN works). This is a critical point: a system monitoring the AGI system must be able to detect this kind of behaviour and, somehow, solve it.

5 Bottlenecks, criticals and food for thought

- How do we teach to separate local network vs external network? We want the AGI to run the protocol code before interacting with the external network: since the AGI will probably be a distributed system this "internal/external" separation may not be so clear.

- Since we don't know the learning technique which will create the first AGI, can we adapt this method to every learning technique? This is not clear to me, it would be great if we find a way to fine-tune the AGI after the learning process: in this way, we can adopt this protocol method a posteriori.

[6] This requires further analysis.

[7] We need a sort of hierarchical goal function: submodels which operate on the "base" need also to do the upper layers. (further clarifications needed).

- Can the AGI split (copies) himself in submodules which don't send outputs in the protocol? Yes, this could happen and it's a problem. Even if, theoretically, the submodules are created to optimize the goal function we can't be certain that each of them will send outputs through the protocol. This is not clear and should be tested but, a naive goal function design will allow the submodules to take care of partial operations, without the need to optimize the entire goal function. Let me explain it with an easy example: let's suppose that the goal of an AGI is to output a pair containing the sum of the input numbers and the product of the input numbers. The AGI could delegate the summing operations to a copied submodule and the product to another. Thus, each module is optimizing just a component of the goal function. Then the AGI can collect and send the results. This is critical, in general, for the alignment problem: even if the AGI is properly aligned, it could split-copying in submodules which may behave dangerously or in an unexpected way. My intuition is that the AGI will avoid this naive split since the submodules may behave against his goal function but further investigations are required. It's probably dependent on the learning technique and the goal function design.

- What if the company lose (or compromises) the private key of the digital signature? We could design a series of protocols with different keys and different levels of security. In this way, we avoid a single point of failure but we also increase the overload on the system (which seems negligible to me).

- Since the protocol is open-source, can the misaligned AGI learn to mathematically bypass the undesired encryption contained by the shut-down update? No, it would need to solve the hard-mathematical problem that we assume to be unbreakable even for super-intelligent AGI.

- What if the AGI system is trained with a different paradigm, where the goal function is not the main optimization target? For example in the Amplification method. There's no clear "outer" and "inner" alignment distinction and no clear goal function to optimize.

- Is this method a possible solution to solve AGI safety? Not at all, it can just contribute toward that direction. To be effective, we would probably already need a decently aligned AGI system. However, it can be a way to safely build feedback to correct the AGI, making possible the "human in the loop" approach.

6 Further work

It's not evident to me how to design a protocol to monitor the AGI system and send a warning to the programmers if the AGI system is going to do something risky. It is fundamental to design and test a monitoring procedure capable of detecting dangerous behaviours. It may also be useful to standardize security thresholds for the monitoring procedure to avoid unfair competition between companies.

We may also want to test if fine-tuning (to send outputs through the protocol and not modify it) the AGI system after the learning process is possible and how to do it effectively. Further work is needed to test this.

Thinking about other advantages that a protocol can obtain: other than safety concerns, the company write a protocol for licensing or stuff. The more appealing the protocol advantages are, the more the company will be incentivized to adopt it (decreasing the so-called ”safety/alignment tax”).

7 Conclusion

In this article, I proposed a new concept to work within the AGI field: freedom. Moreover, by reasoning with this concept, we may find new ways of controlling and handling risks. I also proposed a way to constrain the AGI system and handle detected risky behaviours for humans: the key idea is to force the AGI system to send outputs through a programmed protocol by training it to do so. We can avoid unauthorized modifications by penalizing them in the same way. It can be updated safely (avoiding adversarial updates) thanks to the digital signature of the programmers. The protocol can implement a monitoring procedure and send a warning to the programmers if we spot risky behaviours. Therefore, we can solve the stop button problem by updating the protocol. Numerous bottlenecks and criticals are still open and need further work to be solved.

References

[AZJ06] Hermina Alajbegović, Dževad Zečić, and Hasan Jamak. "DIGITAL SIGNATURE ALGORITHM (DSA)". In: Sept. 2006.

[Soa+15] Nate Soares et al. "Corrigibility". In: *Workshops at the twenty-ninth AAAI conference on artificial intelligence*. 2015.