Bad News for Artificial General Intelligence

https://mindmatters.ai/podcast/ep161/

Robert J. Marks:

Welcome to Mind Matters News. I'm your affectionate host, Robert J. Marks. I recently vetoed a family member's suggestion that we put a lock on our home that could be opened using a cell phone app. I didn't want it. Why? There is just too much that could go wrong. An old fashioned key lock is simple and reliable. I was unsure about cell phone apps and haven't had the best of luck with some of them.

Robert J. Marks:

This is a problem with complex systems. The more complex the system, the more it can go wrong. Artificial General Intelligence or AGI will be complex, for all the stuff it's expected to do, it has to be complex. And as complexity increases linearly, the way things that can go wrong increases exponentially.

Robert J. Marks:

This is especially concerning when human life is involved. We talked about this last podcast with Uber killing a pedestrian and also the Soviet Oko saying that the United States was attacking the Soviet Union with thermonuclear missiles. To talk with us today about this topic is PhD students, Sam Haug, and freshly minted PhD, Dr. Justin Bui. Both are members of my research group and are really, really smart. Sam, welcome.

Sam Haug:

Thank you. Happy to be here.

Robert J. Marks:

You too, Justin.

Justin Bui:

Thanks for having us, Bob.

Robert J. Marks:

Sam, you're the first author on a peer reviewed paper that showed the reasoning behind this exponential explosion of contingencies. Can you explain in a simple way as possible why contingencies increase exponentially as the complexity of a system increases linearly?

Sam Haug:

This explanation is probably best illustrated by example. So here, let's consider a washing machine, and we're going to have a very simple washing machine. All it does is it has two settings, it either washes the clothes for a long time or it washes the clothes for a short time. In addition to those two settings, it only has one singular sensor to figure out how long it should wash the clothes. And this sensor is going to measure how heavy the load is.

Sam Haug:

So in this very simple example, there's only one sensor involved that is keeping track of one variable in our design project. And it yields two possible outcomes, either washing for not long time or washing for a long time. So if this washing machine is to be correctly designed to handle these loads well, all that needs to happen is that the washing machine needs to be tested for a heavy load and a light load. And if it handles both of those scenarios correctly, then you have designed the perfect washer for the design project you have.

Sam Haug:

And in this particular instance in the design process, the assumption is that you'll begin with a prototype, you will test that prototype to see how well it handles the contingencies that you're expecting. And if it handles those contingencies well, you're done with your design process. If it does not, then you'll need to make some tweaks to make sure that it does. And so this is going to be kind of the framework that we use in the paper to discuss complexity of design.

Sam Haug:

Now let's talk about just a slightly more complex example. So we still have the same washer, it is still only able to discern a few variables using its sensors. And now we're going to add one additional sensor, which is how dirty the load is, which is the commonly referred to measure is turbidity. So if it is very turbid load, it's very dirty, and so you'd need to wash it also for a long time. And if it is not turbid, if it's very clear water in the washing machine, then you don't need to wash it for as long.

Robert J. Marks:

And you can do this by simply putting something like an LED light and seeing how much attenuation there is from the light to the sensor. And the more turbid the water, the more attenuation is going to be given. Right?

Sam Haug:

Yes.

Robert J. Marks:

So you've got you've added a sensor. Okay, go ahead.

Sam Haug:

Yes. So here, our design is getting a little bit more complex. And now we have instead of two possible input loads, we have four possible input loads. We could have a light, clear load, we could have a light turbid load, a heavy, clear load, heavy, turbid load. And so now we have increased, we have doubled the number of possible input loads that we can put into our washer. And we'll begin to refer to these possible inputs as contingencies because that is what they end up being in the design process.

Sam Haug:

So in order to now design the perfect washer for this washer that now has two sensors, you need to test four possible loads. And if this washer correctly handles all four of those loads, then you've finished your job, you have designed the perfect washer for this example. And we begin to see here as we add variables, each time we add a variable, and in this case, each of the sensors only has an on or off reading, so there's no scale or range of values. So in this case, every variable you add doubles the number of contingencies that your washer will need to account for.

Sam Haug:

And to give you a little bit of a numerical estimate of what this does, if we increase the number of sensors on this washing machine to 20 sensors, so it keeps track of 20 different variables, so heaviness would be one, turbidity would be one, it could go through any number of other possible examples. With a still very simple system with only 20 sensors, each one can only be on or off, there's no range of inputs for these sensors, there are already over a million contingencies that you would need to design your washer for, which is just incredible.

Sam Haug:

Looking at a little bit more complex system of an image recognition software, for example, one of them would be the wolf and dog classification that we talked about last time where you feed a neural network a picture of either a dog or wolf, and it tells you which it is. If you wanted to fully characterize the performance of this system, you would have to test every single combination of pixels in the image size that it's going to be fed.

Sam Haug:

So for a small 100 by 100 pixel image, that's 10,000 pixels that you need to test. And each of those pixels has 256 gray levels and three color choices, which is the RGB, which is red, green, and blue values for each pixel. In this still relatively small design example, if you wanted to fully test the performance of any image classification software you're designing, you would have to test it 10 to the 29,000 times, which that number is so large, it's difficult to imagine.

Sam Haug:

So as a bit of a ballpark estimate here, the number of atoms in the known universe is estimated to be around 10 to the 80th, which is an incredibly large number. But the number of contingencies with this small 100 by 100 image is just unfathomably larger than that 10 to the 29,000th power, which is just bigger than anything we could probably imagine.

Robert J. Marks:

As we say in Texas, it's bigger in Dallas.

Sam Haug:

That's right.

Robert J. Marks:

It's just an enormous, enormous number. Now, of course, I think that testing all possible images would probably not be wise, and we're going to talk later about how you reduce these number of contingencies by reducing the problem a little bit. So that was an excellent example. Thank you. Thank you, Sam. Software engineers want to design systems like AI to avoid the problems of the contingency explosion that you just talked about. For example, the image, we wouldn't want to do all of the 10 to the big, big number tests. So what are some ways to avoid unintended contingencies?

Sam Haug:

That's a good question. So one of the primary ways that we can mitigate these effects of the exploding contingencies is with what we call domain expertise, which is a designer's very intimate knowledge with the design that he's creating. So for example, in the area of self-driving cars, which are extremely complex, some domain expertise there might be familiarity with traffic laws, familiarity with the physics of acceleration and braking, and turning and such. So domain expertise is just ground level knowledge of the environment that you're going to be placing your design in.

Sam Haug:

In the example of the image recognition design, some domain expertise there might be in recognizing that your image recognition software will not be exposed to random static noise, for example. And so it may not be as important for you to test all of the possible combinations of static noise for your image, but to focus on the images that will probably be presented to your design such as pictures of wolves and pictures of dogs, and to make sure that those are classified correctly. So that's domain expertise.

Robert J. Marks:

Okay. I use this example a lot. I used it in a podcast with Ola Hössjer and Daniel Díaz. But one of the great illustrations of the need for domain expertise is Formula 409. Have neither of you heard of Formula 409?

Justin Bui:

The cleaning solution?

Robert J. Marks:

The cleaning solution. Okay. I asked Daniel, who is from Colombia, and I asked a Ola who is from Sweden, if they had ever heard of it? And they said, no, no, no. They must use something different. Well, the reason the Formula 409 is labeled Formula 409 is it took for 409 experiments in order to design that final result. And that required domain expertise. I'm sure it was done by chemists. I'm sure it wasn't done by junior high students, for example. And in fact, it was done by a total novice, it would be called formula 2,642,000, or something like that.

Robert J. Marks:

So yeah, domain expertise really can be used as a technique to reduce the unintended contingencies and that's what they did. In fact, this is very interesting. We know about Edison testing thousands of different filaments when he generated the light bulb. And Tesla, who was kind of a nemesis of Edison, came along and he dissed Edison. He said, "You don't need to test all these 10,000 different combinations of filaments. If you just had a little bit of book learning, you could get this down to 100 or 200." Because some of the things that Edison was testing, Tesla considered kind of stupid. So that's another example of the need of domain expertise.

Robert J. Marks:

Another one is WD-40 that I like to use, which is water displacement system mastered at the 40th try. And this was done by an industrial chemist, I think his name was Larson. And if he had not had domain expertise, we would be using WD-5 million or something like that. So in order to test these and in order to do good design, if you will, we need this expertise. And that's something which is really, really important. Justin, do you have any thoughts on this?

Justin Bui:

Kind of like we said, the addition of subject matter expertise really does reduce the complexity of things. I think a lot to my research tied lot with image recognition and classification. Some of the techniques that are implemented in a lot of the larger scale systems deal a lot more with traditional computer vision techniques, histogram correction, color matching and correction, image resizing, and .... Kind of, the more you can do on the front end, in the pre-processing side of things, the much more simple your AI system can be. And it aligns quite well with some of the increasing complexities that you all have documented in your paper.

Robert J. Marks:

Yeah. Okay, fascinating stuff. I think I've learned from you, Justin, that there's a lot of standardization of the AI. In other words, there's a sort of conformity that is used in order to sculpt the input to deep learning so you don't have to consider so much. Is that fair to say?

Justin Bui:

Yeah, I think so. In fact, my intuition and my gut feel says that's where a lot of the subject matter expertise is actually best used. Right? If you can get, let's keep running with the example of an image classifier, if you can get the best possible most standard looking data, the most clean, most precise data, the development of your AI system will be that much more simple. You can reduce impacts of noise or color mismatch, lighting variations, all "in the input pipeline", meaning that you can minimize and optimize the implementation of your AI system.

Robert J. Marks:

Yes. So standardization, I guess I look at it as it reduces the contingencies by decreasing the complexity of the problem that you're trying to achieve. You've standardized everything, if you will. Sam, there are some other obstacles facing development of complex AI. We talked about, for example, in jeopardy of Watson repeating an incorrect answer. And those are covered very interestingly by a quote, I think, made popular by Donald Rumsfeld. I'd like you to talk about that for a second.

Sam Haug:

Of course. This quote is given by former State Secretary of Defense, Donald Rumsfeld. And the quote here is, "As we know, there are known knowns, there are things we know we know. We also know that there are known unknowns. That is to say, we know there are some things we do not know. But there are also unknown unknowns, the ones we don't know we don't know."

Robert J. Marks:

The funny part about that quote is it sounds like double talk if you read it real quickly. But if you sit down and examine it, it's really meaningful and applicable to the sort of thing that we're considering.

Sam Haug:

Yes. So we've taken kind of a very esoteric look at breaking down these known knowns and unknown knowns and et cetera. And we've kind of lumped them into four categories. One of them is the known knowns. These are the tests that we have conducted on our design, and we have evaluated the result of them. And so these, we're very sure that these are correct knowns, because we've actually done the testing, we've seen how it performs, and there's not much more to know about these particular performances.

Sam Haug:

The next would be the known unknowns. These are the tests that we have not conducted. And we know that we have not conducted these tests, because we haven't tested it. And so we are aware of our lack of knowledge in these particular environments and these particular circumstances. Another type of these unknowns is the unknown knowns.

Robert J. Marks:

That sounds like an oxymoron, but it isn't. The unknown knowns. Okay, go ahead.

Sam Haug:

Very correct. Yes. So these unknown knowns are things that should be obvious, but have been overlooked by the designer. And so going back to some of our examples that we mentioned in the previous podcast, the example of IBM Watson repeating an incorrect answer that was given by another human contestant. This would be one of those unknown knowns, where the designer, who's watching the contest would give themselves a facepalm, because they know that they should have foreseen this particular contingency, but they haven't. And so these are contingencies that are obvious, but just have not been included.

Sam Haug:

The final classification of the knowns and unknowns are the unknown unknowns. And these are the most troubling situations and circumstances. Because even a designer with expertise in the domain, they did not foresee the possible outcome of this particular circumstance. And so these would be, for example, self-driving cars attempting to classify plastic bags when they're moving and not moving. The designers probably would not facepalm if their car encounters a flying plastic bag that it is unable to classify correctly, because they didn't foresee that. And it's not something that they should have foreseen that was extremely obvious, it was something that just couldn't have been foreseen by a designer with domain expertise.

Robert J. Marks:

Fascinating. I guess the unknown unknowns is really what is a big problem.

Sam Haug:

Right.

Robert J. Marks:

So I think in the Oko example, where incoming missiles were interpreted from the sun reflecting off of clouds, that was probably an unknown unknown that wasn't even considered in the design of Oko, which is unfortunate. Here's a counterargument. We see highly complex systems that operate reliably. An example of that is, you and me, we're human beings, we are put together, we are very complex, but we still seem to work well. Why? What is going on here? And how is that consistent with the theory that we've just laid up?

Sam Haug:

Yes. So I definitely agree that human beings are extremely complex and extremely well made. I personally believe that this is because humans were created by a creator with an extremely large depth of domain expertise, who is able to-

Robert J. Marks:

That is a great phrase, Sam. I appreciate. Our creator has a deep, how did you put, a deep knowledge of domain expertise. That's great. That' funny.

Sam Haug:

So our designer, he doesn't just have expertise of the domain, he created the domain. And that just has an infinite depth of foresight and predictiveness, where he is able to design these incredibly complex systems and foresee all possible events that they will ever encounter in history or in the future, and design a human being who is able to overcome and adapt to a lot of these circumstances.

Robert J. Marks:

Even so, I'm thinking of the design of human beings, we're still not perfect. There are some in our design, I don't know if there are unintended contingencies or not, but things like COVID, for example, you have an adverse effect to that. We weren't designed to handle COVID, especially old people like me, or even something similar, like eating hemlock, the way that Socrates was killed. We also see defects like in birth defects, diseases such as cancer and things of that sort. Isn't this an example of contingencies which we would prefer not to see in the design of humans?

Sam Haug:

Yes. So the way I like to think about how human beings fail in certain circumstances falls into two categories. The first category is that our creator intentionally did not design us to withstand this particular contingency. And this can come from a couple of reasons. One is, when designing a human being or any incredibly complex system, there are some design trade-offs that exists where you can design a human being to be able to resist the effects of eating hemlock, for example, but the cost for doing that may be large.

Sam Haug:

For example, you would need to include an entirely new metabolic pathway to account for that particular poison. And doing that for any number of poisons may just not be feasible in the size of human body. I don't claim to know about all the design implications of making a human being, but I'm sure that there was some level of intentionally not designing human being to withstand some things for trad-off reasons.

Sam Haug:

And then the other category of things that humans fail or the human design does not withstand, would be due to the fall. I believe in the God of the Bible, who designed us perfectly, and we sinned and fell. And as a result of that fall, the perfect design that God has made was corrupted. And all of the contingencies that he has foreseen, some of the mitigating factors to avoid or overcome those contingencies may have been affected by the corruption of the fall. And so that is where I think diseases and stuff of that nature comes from, because I don't believe that those were intended pre-fall for disease and death of that sort.

Robert J. Marks:

Whatever the cause, we do have something in design, engineers know this, called a Pareto trade-off. This is a trade-off between optimal performances. I work my way through my master's degree as a disc jockey. And one of the things we used to do is we used to cut commercials. And sometimes the copy for the commercials came from the people that were sponsoring the commercials.

Robert J. Marks:

And we had one, and I remember it, because it's so hilarious. It was a place called Charlie's Fish Market. And at the time, there was an explosion in price of meat like pork and beef. Anyway, it was the copy. The copy was, good meat ain't cheap and cheap meat ain't good. So eat fish. That was the ad for Charlie's Fish Market. Now, that explains a Pareto trade-off. Now what do we mean by that? In our world now there's a trade off in performance.

Robert J. Marks:

I'll give you an example with cars, safe cars aren't cheap and cheap cars aren't safe. That's just like Charlie's Fish Market ad, right? So what you have to do is you have to do a Pareto trade-off, a trade-off between being a cheap car and a safe car. If you want to safe car, drive around in a Humvee that has extra armor plating on it. And if you want to go down cheap, get a little scooter and don't wear a helmet or something. But you have this entire gambit.

Robert J. Marks:

And it turns out that Pareto trade-off says, "For a certain price, there's the best safety that you can get in a car." I think if the only criteria for buying a car was the safety and the price, that if you're like me, you would set the price and then see the maximal safety that you can get. And so this is inherent in design, at least, that we experienced today. I agree with you, Sam. I don't think it was applicable before the fall. But certainly, today it is. So this is something that we are certainly stuck with. Okay, any final thoughts?

Sam Haug:

I have just a little bit more on how domain expertise can help in the design process.

Robert J. Marks:

Okay.

Sam Haug:

So I did mention that domain expertise can be used to kind of reduce the number of tests that you need to perform on your design. Because there are some circumstances that you don't really care how your design performs because you don't expect it to be put in that circumstance. But another way that domain expertise can help in the design process is by forecasting what the result of a test would probably be.

Sam Haug:

And so this saves a lot of time in doing the actual physical testing because the designer is able to very quickly look at an environment and say, well, I know that it will perform well there or I know that this particular aspect of the environment will cause it to perform poorly. And so that can reduce the number of tests that have to be physically performed because the designer has enough domain expertise to know how it would perform.

Robert J. Marks:

Yeah. The whole design thing is a big iteration, isn't it?

Sam Haug:

Yes.

Robert J. Marks:

You design, you test, and then you redesign. And that's the reason we talk about WD-40 and Formula 409, it was an iterative loop. So not only does the design have to be well, for example, for AGI, the software engineer has to know what they're doing. But there is intelligent testing, where you go out and you need to test the AGI, and then do variations in order to improve the AGI as you find out different places that it works.

Justin Bui:

To build on that too, write the testing and verification is its own area of subject matter expertise, I think one that's often overlooked. It's funny, to give everybody an example of subject matter expertise, so I ordered a new Bronco last year-

Robert J. Marks:

The car, not the horse.

Justin Bui:

Correct? Yeah. The horse probably would have shown up by now. But it's very interesting, because if you've followed along with the release of that vehicle, they had a roof issue for all the hardtops. It turns out that they decided to replace all of the hardtops that were built or previously issued up to, I believe it was August. And when you observe kind of what happened and how it got to that scenario, it turns out that they had some type of QC that permitted faulty hardware to get into the loop.

Robert J. Marks:

QC?

Justin Bui:

Quality control.

Robert J. Marks:

Quality control, okay.

Justin Bui:

And you think about it like, well, from a testing perspective or a verification perspective, that's something that should have been caught, but maybe they just didn't know what to look for. It ties very well into the testing expertise for an AGI system. We talk about the known knowns, the known unknowns and the unknown unknowns being kind of major hurdles. And the unknown unknowns are the most dangerous kind because we don't know that we don't know them.

Justin Bui:

It's one of those things that when you start looking at verifying a system, you could almost argue that requires more expertise than developing it, in some cases. And so I think that's going to be a topic that you see more and more of as we continue to dive into these areas. And you continue to see more and more AI systems deployed in the real world. You get these scenarios like Uber where you strike and kill a pedestrian, and pretty much every engineer is probably sitting there saying, well, okay, what are the circumstances that could have led to this?

Justin Bui:

It's such a complex system with so many different subject expertise requirements that when you when you look at it in an unbiased light, it's quite a bit to overcome. And so I think, kind of to tie things together, AGI is becoming less general and more specific. And I think that's kind of where we'll see a lot of the direction head in the foreseeable future is a lot more specificity, kind of a step away from the general application.

Robert J. Marks:

That seems to be where it's going. Even if we could overcome the exploding contingency problem, there are other obstacles that cast doubt on successful development of AGI. And I have to keep pounding this home because this is one of the major stances of The Bradley Center. In terms of duplicating humans, AI will never be creative, never understand and will never be sentient. And we cover these topics on Mind Matters News. These are additional obstacles, which I believe are not overcomable. Is that a word, overcomable? I think it is.

Justin Bui:

Insurmountable?

Robert J. Marks:

Insurmountable. That's a better word. Thank you. Thank you. So the obstacles of designing complex AI can possibly overcome but it will require a lot of expert engineering. Thank you, Sam and Justin. Our guests today are a PhD students, Samuel Haug and Dr. Justin Bui about obstacles of designing AGI. And so till next time on Mind Matters News, be of good cheer.

Announcer:

This has been Mind Matters News with your host, Robert J. Marks. Explore more at mindmatters.ai. That's mindmatters.ai Mind Matters News is directed and edited by Austin Egbert. The opinions expressed on this program are solely those of the speakers. Mind Matters News is produced and copyrighted by the Walter Bradley Center for Natural and Artificial Intelligence at Discovery Institute.