AI: The Potential and the Problems

https://mindmatters.ai/podcast/**ep193**/

Announcer:

If, like us at Mind Matters News, you keep an ear to the ground for any stories in the world of artificial intelligence, then you've likely heard the ongoing controversy about whether Google's chatbot LaMDA is sentient or not. To make a short story even shorter, it's not. As we've discussed in the past, sentience and other mental qualities like creativity, qualities that would contribute to what researchers call artificial general intelligence, cannot be reduced to any pure algorithmic form needed for incorporation in an AI. This week, we have Samuel Haug and Justin Bui joining us as we discuss the problems with pursuing artificial general intelligence and how difficult it can be to account for everything that could go wrong with such complex AI systems. Enjoy.

Robert J. Marks:

Greetings. This is Mind Matter News, and I am your full-figured host, Robert J. Marks. Isaac Newton was the genius that founded classical physics. He also invented calculus. He also did other things. Newton was a student of the Bible, specifically Bible prophecy, and he wrote extensively on his research. Newton also dabbled in alchemy. Now, most think of alchemy as the quest to turn lead into gold, but there's a lot more to alchemy than doing this. Some in alchemy pursued creation of a so-called homunculus. Homunculus is a little person created in a test tube, kind of like. If you watch the 1935 classic monster movie, the Bride of Frankenstein, you see a scene where the mad scientist, Dr. Pretorius, shows off his homunculi. I think that's the plural of homunculus. He shows off his homunculi to Henry Frankenstein.

Robert J. Marks:

No one to date has created the alchemist dream of a homunculus. And if you exclude maybe cloning, I don't think that they probably ever will. The search for the homunculus today has been replaced by a search for artificial general intelligence or AGI, artificial general intelligence. Terms keep changing in a rapidly evolving field, AGI used to be called hard artificial intelligence. There's some that actually have a split definition, but we're just going to stick with artificial general intelligence. By any name, the search for artificial general intelligence will prove as useless as the search for the homunculus. Not everybody shares this opinion. That includes Elon Musk. That includes Stephen Hawking. But even so, there is a growing evidence AGI will never be achieved.

Robert J. Marks:

What does AGI do? AGI seeks to duplicate and exceed what you and I do. If artificial general intelligence has achieved, some say we will become pets of computers. There are some who worry that AI will begin to write better and better AI. The point where AI becomes superior to humans is called the singularity by Google's Ray Kurzweil. If this happens, watch out, AI will write better software that writes better software that writes better software in an endless staircase of ever increasing intelligence. And there are smart people who believe this will happen.

Robert J. Marks:

But AGI is not happening and there's growing evidence it never will. AI can be written to mimic many human traits, but there are some human characteristics that will never be duplicated by AI. We cover this a lot on Mind Matters News. Properly defined, these include these properties that will never be achieved, include creativity, sentience, and understanding. In fact, AI seems to be going in the opposite way. More and more human expertise is being folded into the AI software. The added intelligence in AI is not due to AI, but is due to human creativity and ingenuity infused in the software by the programmer.

Robert J. Marks:

To talk about these things, our guest today is Dr. Justin Bui. Justin is a freshly minted PhD from my research group at Baylor University, and he specializes in among other things, artificial intelligence and deep learning. Justin, welcome.

Justin Bui:

Yeah. Thank you very much for having me on the show, Bob.

Robert J. Marks:

Great. Before we go into some trends in artificial intelligence, what I'd like to do is describe the playing field that you have been watching. I think anybody that is involved in the development of artificial intelligence is familiar with these resources. There is, of course, the literature, and there's a vast literature in AI, but in computer software, there's a heck of a lot more incredible resources. There are incredible resources available on the web, many related to AI. And the incredible part is, most of these are free. So first, let's talk about the software, Justin. AI software is widely available. It's free and it's powerful. It's available to anyone on the dot. Could you go through some of the AI software and some of the things that this AI software that's available for free does?

Justin Bui:

Yeah, sure. It's an interesting playing space. It seems like every day there's a new tool that comes out, that makes everybody's lives just a little bit easier. The big ones of course are PyTorch and TensorFlow, both being driven by Facebook and Google respectively. They make up, I believe it's upwards of probably 75 to 80% of a lot of the machine-learning systems out there, if not more. They're very easy to use. It's tremendous development. And the amount of available resources associated with these tools is phenomenal. And kind of going hand in hand with that is the use of free web resources. A lot of systems out there provide free computational resources, basically virtual machines that anybody can sign up for and use. They can design, deploy, evaluate any machine-learning model that they would like. It's actually quite interesting to see how prevalent some of these systems have become.

Robert J. Marks:

Yeah, this is really interesting. One, one of the fascinating things is the free available computation. AI, like deep neural networks, for example, can take a long time to train. So you're crunching the computer again and again and again, and yet there is available fast software resources available on the web to allow you to do this in the cloud. That to me is just amazing, that people are making this available for free. There's also something called fast AI. What is fast AI?

Justin Bui:

So fast AI is kind of a wrapper for PyTorch with a lot of pre-built models. It's meant for rapid proof of concept testing, if you will, takes advantage of a lot of transfer learning techniques, where just a normal, we'll call them, everyday person, but really anybody, can pick up a Jupyter Notebook or a little bit of Python code and follow along on one of their tutorials and effectively deploy a classification model or regression model. It's really meant to help speed up the initial proof of concept for a lot of these model development processes.

Robert J. Marks:

It's kind of an interface in a way, is it right?

Justin Bui:

Yeah. Yeah. I think a good way to classify would be like a high-level wrapper almost. But again, it lets you take advantage of some of the work that's already been done, and that ultimately cuts down on somebody's development cycle.

Robert J. Marks:

So PyTorch, the Py is for Python, Python is a computer language that's available for free. Everybody can use it, right?

Justin Bui:

Correct. Yeah. Torch itself is actually built on Lua, which, for those that are perhaps more intimately familiar, is a scripting type language. And so PyTorch is, you're right, the Python high-level wrapper for the Lua interface, that is torch.

Robert J. Marks:

Okay. What sort of stuff can you do with all this free software, specifically some of the stuff we see in the news today?

Justin Bui:

Oh yeah, sure. I mean, all of these tools have high-level code wrappers for doing custom layer developments. Of course, you've got convolution layers, which, for those that are familiar, go into convolutional neural networks. You have transformer layers, which are gaining popularity.

Robert J. Marks:

Could I interrupt you just for a second? What is a wrapper, for the general audience here?

Justin Bui:

Good question. So a wrapper is kind of just like a high-level function call. It's a chunk of code that ultimately makes deploying something more complex, very easy. You could think of it kind of as a super function in a way.

Robert J. Marks:

I see. So you might have software and you go build the pyramids, and you click yes, and the pyramids are built, something very, very big happens.

Justin Bui:

Exactly. It'd be something like build a pyramid.

Robert J. Marks:

Okay.

Justin Bui:

And all the hard stuff is done underneath the hood, so to speak.

Robert J. Marks:

Okay. So you were talking about some of the stuff that you can do with all of this free available software and all of this free available computational space.

Justin Bui:

Yeah. Yeah, of course. Like I mentioned previously, you have convolutional layers, you have recurrent layers, which are things like LTSMs. Now, that add a little bit of memory, so to speak, to the neural network transformers, as I mentioned previously as well, and a whole bunch of combinations in between. These high-level layer calls, if you will, it really lets you get creative with the architecture. You can combine different techniques into this. I guess, you can consider it an amalgam of different neuron types with different inputs and outputs. And you can kind of create this hydro looking system, if you will, where it can take various inputs and create various outputs. It's really great because it lends itself to this creative model development through its flexibility. And both of these tools allow that to happen. You kind of see this competition back and forth. It's been interesting to follow along as these tools develop.

Justin Bui:

When I started doing a lot of my research, TensorFlow 2.0 was still relatively new. I believe it was still in beta actually. And most recently, I believe they're up to stable release 2.6. PyTorch, I believe, in a similar fashion, was at about 1.2 at the time when I started my research. And most recently, their stable release is 1.9. So you're seeing some pretty heavy iteration improvements in these tools. It's great because it's driving a lot of the AI and machine-learning development kind of going hand in hand with deployment of these tools, as you're seeing more and more of these free resources that you've mentioned before becoming available. I kind of view them as a mix of things. One, it gets the tools and the hands of people to experiment and to learn.

Robert J. Marks:

Now, the interesting thing, this is available to anybody in the world.

Justin Bui:

Yep.

Robert J. Marks:

All of our adversaries in the United States, at least politically, militarily like China, Iran, can plug in, get this free software and do all this artificial intelligence and do it all for free.

Justin Bui:

Yeah. That's right. It's kind of a double-edged sword in a way. But I think in an ideal world, anyways, what you're doing is, you're providing the masses with the tools and the opportunities to kind of push the envelope forward. I think it's a good thing because it makes the accessibility and the learnability of the techniques much more grounded, whereas before it was pretty heavily academic and very computationally intense and required a lot of subject matter expertise. A lot more of the innovation now is who can get to the finish line first kind of deal. It should, in a way, encourage some more competition.

Justin Bui:

There is one caveat, of course. The free resources is that they are constrained. Most systems you're typically limited to a fixed number of training or running hours. You get a fixed amount of memory, which if you think about, I think most systems provide between two and 16 gigs of RAM to use, which sounds like quite a bit. I mean, most people probably have 16 or 32 gigs of RAM on their personal computers. But if you're loading a data set that contains, say, 150 gigabytes of DICOM data, for example, or other medical data, well, that's not going to fit in memory. And you'll find out very quickly that these systems break.

Robert J. Marks:

However, if you do have the resources of computation and memory by yourself, you can download the software and run it in your system with basically limitations, which are totally dictated by your resources that you have locally. Right?

Justin Bui:

Yeah, that's correct. That's one of the nice things about the open source tools, is that if you want to build yourself a small super cluster with a couple terabytes of RAM and a whole bunch of processors, you're of course welcome to do that. You have almost no limitations other than making sure that you have compatible drivers and that all of your system software plays nicely.

Robert J. Marks:

I was thinking of some of the specific things of the news that could be done with this software. One of them is deep fakes. Can you do deep fake images and videos with the software?

Justin Bui:

You can. Yeah. In fact, there's a system out there called Kaggle, which I believe we'll probably mention again.

Robert J. Marks:

Oh yeah. Let's talk about that. What is Kaggle?

Justin Bui:

So Kaggle, it's actually owned by Google. I believe they've acquired somewhat recently. It's a open source platform that provides computational resources to data scientists, machine-learning engineers. But of course, anyone has access to it. If you have an email, you can get access to it. It's a website that allows people to post competitions. So there are a lot of design competitions of various types. There's image classification, stock price prediction, housing price prediction, a couple different things that kind

of just highlight the industry in general. They kind of have modernized the open source resource sharing community driven AI push, if you will. And it's been very interesting . I've spent some time perusing on their forums and reading through the discussions and taking a look at some of their competitions. People get very creative. It's kind of fun to watch as a spectator and see how people approach problems and what techniques do they try. If things aren't successful, do people share that experience or do they kind of brush it under the rug and move on?

Robert J. Marks:

I remember from, gosh, a long time ago that Netflix put out this competition to come up with software that, when given user data such as data from you or me, could figure out the sort of things we would like to watch. In other words, things to suggest for us to watch. And they offer big cash prizes for that. Kaggle does this also, doesn't it?

Justin Bui:

Yeah. It's interesting because they're more of a host system where anybody could throw a competition up there. One example is, the NFL has a helmet detection competition going on right now. It's in cooperation with Amazon Web Services.

Robert J. Marks:

Okay. Wait, NFL helmet detection?

Justin Bui:

Yes. Yeah. So what they're trying to do is develop a system that can detect and track helmet locations for players. And what they're really getting at is being able to detect illegal hits, like targeting, for example, by tracking helmets and detecting when there's helmet-to-helmet collision. So part of it is player safety, but they're looking at ways to automate this, because if you think about a human in the system, a referee has to watch how much of the field, well, really all of it. So they miss some things from time to time. And when you think about it from a player safety perspective, you want to be minimizing or ideally completely eliminating some of those rough shots. And the thought is that if you developed an AI system to be able to do that, you could shift that burden, so to speak.

Robert J. Marks:

I see. So this is an ongoing competition right now. Do they supply data to train the neural network or the AI with?

Justin Bui:

Yes, they do. AWS has provided several gigs of video files, of image files. They've even provided some example code from previous competitions. The prize is a hundred thousand dollars split across, I believe it's the top eight, but it's a pretty large prize purse. I think if you follow the competition's history, because this is several years in the running now, they wanted a system that could do everything, but they pretty soon realized that getting a system that could do everything was pretty challenging. So they said, "Okay, let's dumb down the problem. Let's start with helmet recognition and helmet tracking." If you can start with that, eventually you could build up to a system that could detect helmet-to-helmet collisions or stuff like that. And so it kind of hearkens quite nicely to the AGI competition. Right? So I think the thought was that the system would create this kind of master referee that could watch every

player on the field, track locations, detect illegal hits, et cetera, but people are realizing, "Well, turns out that's a lot harder than we thought."

Robert J. Marks:

I can also see this being used by people, such as neuroscientists, to study the impact of these collisions on brain development. We had a guest in a podcast a while back named Yuri Danilov, who was a neuroscientist, did just fascinating work. He said, his indication was that all football games were just terrible and he refused to let his kids play football until his oldest son finally did get on a team. And I said, "Well, what happened? I thought you forbade it." He said, "I was outvoted." So his kid literally played football. But I could see tracking this in real time would be really interesting because you could measure, for example, the acceleration of the helmet. You could do the...

Robert J. Marks:

Let me get a little nerdy here. I think in beginning physics, everybody talks about distance, velocity, acceleration. And then I learned, when I was working for Boeing, that each one of those is related by a higher derivative in calculus. So you start with the distance, you get the velocity, you get the acceleration. And then what is the derivative of acceleration is something called jerk. And if your acceleration changes really quickly, you have a jerk associated with you. And I could see being used with a AGI in order to monitor jerk, which I think that neuroscientists would find very interesting in terms of tracking potential brain damage. And then the cool part is, the derivative of jerk is snap, the derivative of snap is crackle, and the derivative of the other one is pop. That sounds really, really strange, but they could also monitor snap, crackle and pop. But the prize for this is a hundred thousand dollars. That's not minimal. Who is involved in this? Is it universities? Is it companies? Is it both?

Justin Bui:

That's the one thing about platforms like Kaggle, is it really is anybody, anybody who wants to participate can join. So I think from my observation, it's a lot of individuals, or you can actually join teams and coordinate across the world really, if you'd like. There's several teams that are multinational. But it's really anybody who's open to it. And I think the larger thing to take away from that is, it's crowdsourcing the development, so to speak. So you can, in a way, fork up what sounds like a pretty significant amount of money but in the grand scheme of things, from a company perspective, is relatively small, and get basically unrestricted access to the IP that's developed basically for cheap. You know?

Robert J. Marks:

Wow. That is really interesting. These are companies which are kind of, if you will, outsourcing their R&D to competitions and probably getting results a lot cheaper than hiring a bunch of experts in trying to tackle the problem locally.

Justin Bui:

Yeah, exactly.

Robert J. Marks:

Wow. So that works very, very well. One of the things you mentioned to me, Justin, which I appreciate... By the way, Kaggle is K-A-G-G-L-E. It's kaggle.com for anybody that wants to take a look at it. You

mentioned to me that in monitoring these things on Kaggle, that you saw not an advancement of AGI but, in way, a kind of reversal of the AGI. Could you repeat what you told me about that?

Justin Bui:

Yeah, sure. I think to summarize it, what we're seeing is, like you said, it's a 180. You're really seeing almost this hyperspecificity in a lot of the applications. If you go through and you observe a lot of the competitions that have closed, where many of the competitors have shared their code, you see a lot of evidence of transfer learning. So of course, there's some network reuse and stuff.

Robert J. Marks:

Wait, just elaborate just a second on transfer learning.

Justin Bui:

Oh yeah, sure. So with transfer learning, you take an existing system as an existing neural network, and you basically discard some of the weights and biases. So you take some of the train network and you let it forget some of the information, and then you apply it to a new data set. It's really common in object detection and image classification networks, where with these very deep neural networks, you have, say, maybe the bottom four or five layers. The one's closest to the output, they wipe their memory, so to speak, and train it on the new data. And so you have the core detection layers up top, which have been trained and tested and verified, being reused but on a new set of data. It's a pretty common technique and it produces some really great results. And that's one of the things that you see a lot with some of the Kaggle competitions, is a VGG or a resonant being used for top layers, then maybe a little bit of customization on the bottom side.

Robert J. Marks:

Here's the way I kind of understand transfer learning. Suppose that you had a neural network that was trained on dogs, that you trained this neural network to detect dogs. And you would have to spend a heck of a lot of time figuring out this neural network and training this neural network to recognize dogs. Now you want to come along and you want to classify cats. Well, it turns out that classifying cats is kind of similar to classifying dogs. So why would you have to go back and start again at scratch? Why couldn't you use part of that dog neural network to train the cat neural network? And the art of doing that is referred to, I believe, as transfer learning. Is that fair?

Justin Bui:

Yeah, that's a great example. It's one of those things that a lot of people are like, "Hey, why reinvent the wheel when I have a system that gives me 85% of a wheel?" So yeah, you're spot on.

Robert J. Marks:

Okay, good, good. Despite all of these challenges with AGI and your observation that is kind of going the other way, maybe we're waiting for a new theoretical breakthrough, which I don't think will ever be achieved. But nevertheless, there are people that believe that we are making steps towards AGI. And there are those that believe that indeed, this is going to happen. Now, George Gilder, who is one of the co-founders of Discovery Institute and just a genius in terms of economic and business commentary and forecasting, says that this dream that these software engineers have is something which could be called rapture of the nerds, I like that, because it takes a lot of faith to believe that we're going to get there.

Robert J. Marks:

One of these companies, which is just overtly into promoting this, is OpenAI. That's the company that bought us GPT-3, and they claim they are pursuing AGI. I looked up their mission statement and it included the following, this is a quote, "We will attempt to directly build safe and beneficial AGI, but will also consider our mission fulfilled if our work aids others to achieve this outcome." So they definitely believe in this. They have faith that computers will eventually develop AGI. I always thought that was very interesting.

Robert J. Marks:

Now, you and I had talked a little bit about why these software engineers believe that AGI is achievable. I mean, these are guys which are really, really intelligent and they believe that the AGI is indeed achievable. And one of the reasons, I think, is because in terms of AGI operations, such as understanding and creativity and sentience, that they don't understand is not algorithmic. They haven't gotten to the computer science. And the word that I used for this was a so-called keyboard engineer. These are people that, when they're looking for a solution, don't sit down and look at the theory, rather, they go directly to the keyboard. You had some interesting comments on that. Could you elaborate on that?

Justin Bui:

Yeah, sure. It's one of those things that some of my colleagues and I have jokingly referred to as Stack Overflow engineers. It's a very similar concept, but it's a-

Robert J. Marks:

Okay. Stack Overflow, that's a website, right?

Justin Bui:

Correct. Yeah. It's a forum where people can post errors or issues that they're having with their code. It's kind of a community-sourced solution house, if you will. But it's pretty funny because some of the colleagues I've had throughout the years have joked about, "Okay, hey, we just got this problem. Let me go check Stack Overflow really quick." Chances are, somebody's done it before, I'll just reuse it. And so I think that feeds into some of the AGI belief as well, "Oh, well, Open AI has produced X, Y, Z neural networks," and, "Oh, hey, Google and Google's brain team have published on ABC works." If we can start merging these together, the system will just become kind of super intelligent. So I think in some regards it's fed a lot by what people are observing from the major companies and some of the major influencers, like you had mentioned, Elon Musk.

Justin Bui:

Before, you have some very popular people that are promoting, dare I say, preaching, some of these ideas and beliefs. And people kind of latch onto that. It was funny, because when I think of AGI, I kind of think of HAL 9000 or Skynet. Or for those of you that are more into movies, more recent, Altron. These systems that seemingly have limitless resources and infinite knowledge and obviously evil intentions. I think that's one of the things that helps capture people's attention in their-

Robert J. Marks:

Oh, sure. Yeah.

Justin Bui:

... creativity as well. But I think at the end of the day, Bob, like you said, people, they go straight to the keyboard. They don't sit down, think about how to approach a problem, how do we solve it from the theory perspective, and then start deploying it. It's really more, "Well, okay, I need to go make a classifier that tells me the difference between kumquats and giraffes," and they just sit down and start coding.

Robert J. Marks:

And so they import these things and download this software and use this software kind of as a black box, without looking at the deeper theory of how it is created and the computer science of where it came from and the possibilities of doing AGI in the future. They don't address some of the things we talk about on Mind Matters News. They don't address the Lovelace Test for creativity, which has never been demonstrated in artificial intelligence. They don't talk about even simple counter arguments, like Searle's Chinese room, about understanding. And as a result of this, I don't know, we're guessing here, aren't we?

Justin Bui:

Yeah, in a way.

Robert J. Marks:

Yeah. We're guessing, but it seems to me that they possibly don't understand, or at least maybe they've just blocked out this idea that AGI can't be achieved. Really great points. Okay. Any final comments?

Justin Bui:

Well yeah, actually, I did kind of want to build a little bit on that too. I think in some regards, AI and machine learning, they've become catch phrases throughout the world. I used to joke that AI is very similar to the word synergies in the business world. Right? Synergies, everybody wants synergies. The new thing is, everybody wants machine learning. They don't necessarily understand what it is. Like you had said, it's a black box. Let's wave our hands over it. Let's see some results. Are they the results we want to see? Great. We now have machine learning. It's not that easy.

Justin Bui:

But I think a lot of the drive and a reason why a lot of keyboard engineers have a lot of success in their careers and gain a lot of influences, they're able to produce those results, which businesses see and they like. It kind of feeds into this system of like, "Okay, hey, this person's achieved these results. This company's using machine learning. Well, maybe our company can use machine learning. Let's go do the same thing." And the focus, all of a sudden, becomes the material goal. Right? A little bit less about, "Let's create a perfect system," more about, "Hey, let's create this system that provides us the best benefit.

Justin Bui:

Yeah, I think that that's one of the things that really feeds into the keyboard engineer mentality as well, is sometimes you don't get the freedom to sit down at a white board and say, "Okay, hey, how do I approach this problem from a theoretical standpoint, from a high-level concept standpoint, before I

start writing code?" Typically, it's, "I was just handed in assignment. I've got two weeks to do it. I'm going to go to my keyboard and start writing some code."

Robert J. Marks:

Okay. That's fascinating. I think there's a little bit of walk back on the idea of AGI. I did a interview with George Gilder who is neighbors with Ray Kurzweil, and good friends. Ray Kurzweil, of course, is the one that introduced the idea of the singularity and was a big proponent of AGI. And Gilder says, he's noticed recently that Kurzweil has begun to backtrack a little bit on AGI and its implementation. And the problems... I don't know, there's a bunch of problems associated with AGI. Number one is, in the arguments about it, people are using seductive semantics. They say AI will be creative or have understanding or be sentient without really defining what these mean. They're using seductive semantics. And in order to discuss those, you have to be careful in defining them.

Robert J. Marks:

So yeah, we'll see what happens. I have little faith that AGI will ever be achieved. What I was going to say is that AGI has to be defined. And the way that we're defining is, the ability to duplicate what human beings do. There will be a lot of stuff artificial intelligence will do, which is a lot better and a lot more impressive than humans would be. Heck, that happened when they came out with the calculator. Calculator does a lot that I'm unable to do and does it much more quickly.

Robert J. Marks:

We're here to talk about AI. AI design ethics requires AI to do what is designed to do and know more, but problems pop up in complex systems, including any attempts at generating artificial general intelligence or AGI. AGI, whether you think it'll be achieved or not, will by necessity be complex. And the more complex the system, the more that it can go wrong. To talk about this today is our guest, PhD student, Samuel Haug; and freshly minted PhD, Dr. Justin Bui. Both are members of my research group and are really smart. I really feel fortunate to have worked with them and to continue to work with them. So Sam, welcome.

Sam Haug:

Thank you. Happy to be here.

Robert J. Marks:

Okay. And you too, Justin.

Justin Bui:

Yes, thank you very much for having me on.

Robert J. Marks:

Okay. I want to start out with something that rings of Paul Harvey's The Rest of the Story. Either Sam or Justin, have you ever heard of Paul Harvey?

Sam Haug:

I have not.

Justin Bui:

No, I have not.

Robert J. Marks:

Okay. That shows I'm a senior citizen here. Paul Harvey had a series on the radio, very popular series. In fact, wrote a couple of books too, where he recounted a sometimes familiar story and then added a little twist at the end, kind of an Alfred Hitchcock twist at the end, which few, if anyone, had ever heard about. The twist at the end was the rest of the story. It was a little elaboration on the story that nobody expected. And we're going to do this today with some popular AI stories. The twist is going to be something not well known about how AI failed. These failures called unexpected contingencies are our rest of the story. And so it will illustrate some of the shortcomings of AI and illustrate this idea of unintended contingencies, which we want to talk about in the podcast.

Robert J. Marks:

We'll start out with some simple examples and then we'll get more serious cases involving human life. The list is from a peer-reviewed paper that Sam and I wrote with Bill Dembski. It's a peer reviewed paper in the IEEE Transactions on Systems, Man, and Cybernetics. And we'll make a link to that available in the podcast notes. So let's do the following. I'll tell a story, and Sam, I'd like you to give the rest of the story, give the stories twist at the end. Is that okay?

Sam Haug:

Yes. Yes, it is.

Robert J. Marks:

Okay. Okay. First one, number one, Jeopardy is one of the most popular quiz shows in the history of television. Could AI win at Jeopardy? Well, it made big news, the answer is yes. In 2011, the world champions in Jeopardy took on an IBM computer program named Watson. Watson didn't respond to every answer correctly. It wasn't designed to do so. But in the end, playing the game Jeopardy, Watson recorded a resounding win over both of these other Jeopardy champions, and that made headlines, but people left out maybe a little quirk in Watson. So Sam, what's the rest of the story on this?

Sam Haug:

Yes. So in this particular contest, there was quite a funny occurrence where Alex Trebek asked one of the contestants a question, and the answer that the human contestant gave was "What are the '20s?" as the answer to that question, which was noted as incorrect by Alex Trebek. And immediately afterwards, Watson buzzed in and gave the exact same response, "What are the '20s?" And obviously, this answer was incorrect because it was just revealed to be incorrect. And this was something that the programmers of Watson did not foresee.

Robert J. Marks:

Yeah, this was an unintended contingency. I imagine, when the Watson programmers heard this duplicate response, they face palmed and they go, "Oh my gosh, that was such an obvious thing we could have put into the software, but chose not to do." Just fascinating. Watson had great plans for itself in the field of medicine after it premiered on Jeopardy, but it failed. The idea was this, there are just hundreds and thousands of different papers published in the medical field. And wouldn't it be wonderful

if Watson could mine all of this data, which was published in the medical field, and then based on a query from a physician who gave symptoms and details about the case they were dealing with, was able to respond with a list of papers relevant to what was happening? This would save the doctor from wading through thousands of papers in the literature.

Robert J. Marks:

Watson contracted with medical research group and hospital MD Anderson. But after a while, MD Anderson just fired Watson. It just wasn't doing the job. And in fact, we listed this as the number one in the Top Ten AI Exaggerations, Hyperbole, and failures in the year 2018. We listed this on Mind Matters News. Since then, IBM Watson's application expectations have even fallen further. And so we're not sure what's in the future for Watson, but we can see that even though it was working well, it did have this unintended contingencies.

Robert J. Marks:

Okay, example number two for the rest of the story, and this has to do with also another IBM piece of software. In 1997, IBM's Deep Blue software beat world champion, Gary Kasparov, at chess. This made world headlines. One of Deep Blue's moves was particularly curious. The unexpected move psychologically threw Kasparov off his game, and he lost. Kasparov looked at the move and said, "I can see no reason for why IBM Deep Blue made this particular move," and it blew him off his game psychologically.

Robert J. Marks:

One of the chess experts who were commenting about the game said, "It was an incredibly refined move of defending while ahead to cut off any hint of counter moves." Well, I guess skill in a game is like interpreting art in a painting. Some people will look at a painting, and some will think this is great art, and others will say, "This looks like a kid's finger painting." And that was indeed the case for this incredibly educated commentator. So the interesting thing is, what is the rest of the story? Sam, could you kind of finish this out? What is the little twist on this Deep Blue move?

Sam Haug:

Yes. This one's also a little humorous here. It turns out that over a decade after this match, one of the computer scientists who designed Deep Blue, Murray Campbell, he confessed that the move that Deep Blue made, that threw Kasparov off his game, was a random move that Deep Blue had chosen because Deep Blue was unable to choose a good move, and so he just chose one at random.

Robert J. Marks:

Yeah, that's fascinating. I think one of the quotes for Murray is, "Kasparov had concluded the counterintuitive play must be a sign of superior intelligence. He had never considered that it was simply a bug in the code." That was just a fascinating sideline of the rest of the story about Kasparov being beaten by IBM Deep Blue.

Robert J. Marks:

Okay, third story for the rest of the story: A deep convolutional neural network was trained to detect wolves. Now, deep convolutional neural networks, they do make mistakes in their classification. That's just the way that it works. But this one incorrectly classified a Husky dog as a wolf, and so the designers

of the code went in and did some forensics. They found out that this was a fluke of the neural network. What happened here, Sam? What was the rest of the story?

Sam Haug:

Yes. This seems to be a theme of some humor in these stories. The neural network in this particular instance had not been training on the features of the animals that it was classifying, but it had picking up on the fact that all of the wolf pictures that it was fed as training data had snow in the background and all of the dog pictures that it was given as training data did not have snow. And so the neural network had not learned anything about the features of these animals, but had just learned to detect the presence of snow.

Robert J. Marks:

That is really incredible. Justin, have you found out that this is something that which can happen in deep convolutional neural networks? Have you ever bumped across that?

Justin Bui:

Yeah, it is pretty comical when you see things like that. It kind of goes hand in hand with how the network's developed and how it's trained. It takes some very careful thought and preparation to not only design a neural network, but to train it. In fact, it's often said that the 90% of a system's value is in its training and input data. Right? Data is everything. Garbage in equals garbage out.

Justin Bui:

Yeah, it's funny, I chuckle when I hear about that story. But just for kicks, a couple weeks ago, I built a simple little convolutional neural network to classify cats and dogs. It did so with really good accuracy. I think it was in the order of 97, 98%. And then for last, I fed it a piece of fruit. I fed it an image of a kumquat. And yeah, it turns out kumquats are a lot like dogs apparently. So there's just some oddity, some peculiarities that go into developing these systems. Again, garbage in is garbage out. And if you're not thinking about some of these contingencies, you may never come across them.

Robert J. Marks:

That's incredible. Okay, thank you. Okay, story number four for the rest of the story that we're talking about: Self-driving cars are still under development and despite promises are still very far away from a Level 5, which a Level 5 is a self-driving car doing what a human can do. So self-driving cars in early development were trained to watch out for things like pedestrians, deer, and road debris. You don't want to hit a pedestrian. You don't want to hit a deer. You don't want to run over road debris. This worked out most of the time, but there were some serious flaws, at least in this early development. So Sam, what's the rest of the story here?

Sam Haug:

Yes. So this one, there's a very serious side effect that happened in 2018. An Uber self-driving car in Tempe, Arizona actually struck and killed a pedestrian because it was unable to correctly classify this pedestrian as a pedestrian. And as such, did nothing to avoid the collision. One of the engineers that worked on this self-driving car thinks that the vehicle was able to see the pedestrian, but that it was not able to correctly identify it and avoid it. And it's just a very, very sad occurrence of an unexpected contingency.

Robert J. Marks:

So I think the bottom line here is, when AI involves human life and the potential death of a human being, you have to be very, very careful about unintended contingencies. I also think that early in the development of self-driving cars, that blown plastic bags were often interpreted as deer and stationary plastic bags were sometimes considered road debris. And so these are things which can be fixed. We still have hope that this artificial intelligence that caused this death of this pedestrian in the Uber self-driving car can be corrected. But still, this was a terrible unintended contingencies, and they remain a major obstacle in the development of Level 5 self-driving cars. Justin, do you have any comments on this?

Justin Bui:

Yeah, it's one of those things that I think the self-driving nature of cars is still quite a ways away. There's a lot of systems out there that can reasonably identify pretty much every road hazard with a high level of confidence. But when it comes to human life, it's one of those things that even a three, 4% chance of misclassification is catastrophic. So I think a lot more due diligence needs to be paid to classification and detection systems. And it's something that I think is just going to take some time to tackle.

Robert J. Marks:

Yeah. Tesla keeps coming out with all these press releases that they're doing great things, and they clearly are doing great things. One of our writers at Mind Matters News, Jonathan Bartlett, comments extensively on Tesla's update. And I've talked to some people with some Tesla self-driving cars. They can take their hands off the steering wheel for a while, but Tesla will warn them after a while. It says, "Your hands haven't been on the steering wheel for a while. Let's see them." And so they're not ready to go to totally autonomous self-driving cars as of yet.

Robert J. Marks:

Okay, here is the fifth story. The stories are getting more and more serious now. We started out with little things like Jeopardy having IBM Watson repeat an answer. That was a little curious thing. We just got done with talking about how Uber, the self-driving car, would kill people. And now we're going to get to something which is very serious. It's a complex system that could have caused millions of deaths. Let me give you an example, or let me give you the story, I should say. During the height of the Cold War, the US and the Soviet Union were existing on the political knife edge of something called mutually assured destruction or MAD. The idea is that, if the United States blew up Soviet Russia, then Soviet Russia would blow up the United States. And both of the countries would be flat and glow in the dark.

Robert J. Marks:

In order to play this terrible game a little bit more intelligently, the Soviets deployed a satellite early warning system called Oko, O-K-O. And Oko's job was to watch for incoming missiles fired from the United States. On September 26th, 1983, Oko detected incoming missiles. At a military base outside of Moscow, sirens blared, and the Soviet brass was told by Oko to launch a thermonuclear counterstrike against the United States. Doing so would result in millions being killed. The officer in charge, Lieutenant Colonel Stanislav Petrov, looked at these incoming missiles, and he felt that something was fishy. It just didn't feel right. The United States would not launch a preemptive strike doing this sort of strategy. So after informing his superiors of his hunch that Oko was not operated correctly, Petrov did not obey the Oko order. Upon further investigation, Oko was found to have mistakenly interpreted sun reflecting off of clouds as incoming US missiles. In other words, these signals were simply the sun reflecting off of

clouds. There was no US missile attack and Petrov's skepticism of Oko's alarm may have saved millions of lives.

Robert J. Marks:

So we've gone from the very innocent to the very serious of what happens with AI unintended contingencies. Unexpected contingencies from complex AI can become more and more serious as we've seen. I don't know about you guys, but I play Alexa. And when you can't get Alexa to play a song you want, it's annoying, but it doesn't cost any human lives. On the other end of the spectrum, killer self-driving cars and detectors of thermonuclear strikes can't be allowed to make mistakes. If they do, lives will be lost. In the examples that Sam and I have gone through, we have run the gamut from the very innocent of the very serious. The name of the paper, which this is outlined in is called Exponential Contingency Explosion: Implications for Artificial General Intelligence. It's by Sam, William Dembski, and me. It appears in the peer-reviewed AI journal IEEE Transactions on Systems, Man, and Cybernetics. And Sam is the first author.

Robert J. Marks:

Now, in that paper, we also do a bit of math. We showed that the number of contingencies can increase exponentially with respect to the system complexity. The number of contingencies can become so numerous that they cannot all be looked at individually. This is troubling. This is not good news for AGI, which by its very nature must be very complex. We'll explore this exponential explosion of contingency increases as complexity increases linearly next on Mind Matters News.

Robert J. Marks:

I recently vetoed a family member 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 was 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. 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 that's expected to do it has to be complex. And as complexity increases linearly, the way things that can go wrong increases exponentially. This is especially concerning when human life is involved. We talked about this with Uber killing a pedestrian, and also the Soviet Oko saying that the United States was attacking the Soviet Union with thermonuclear missiles.

Robert J. Marks:

Sam, you are 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. This sensor is going to measure how heavy the load is. And so in this very simple example, there is 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:

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 is 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 it sensor. 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 of the water, the more attenuation is going to be given. Right?

Sam Haug:

Yes.

Robert J. Marks:

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

Sam Haug:

Yes. So here, our design is getting a little bit more complex. Now, 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. We'll begin to refer to these possible inputs as contingencies, because that is what they end up being in the design process. 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.

Sam Haug:

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 an off rating. 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. And to give you a little bit of a numerical estimate of what this does, if we increase the number of sensors on this washer machine to 20 sensors, 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-

Robert J. Marks:

Wow.

Sam Haug:

... which is just incredible. 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 a 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. So for a small 100 by 100 pixel image, that's 10,000 pixels that you'd 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.

Sam Haug:

And 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. 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 a hundred by a hundred image is just unfathomably larger than that, 10 to the 29000th 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:

Yeah, 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 in order by reducing the problem a little bit. So that was an excellent example. Thank you. Thank you, Sam.

Robert J. Marks:

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 breaking and turning and such. So domain expertise is just ground level knowledge of the environments that you're going to be placing your design in.

Sam Haug:

And 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. 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 Diaz, but one of the great illustrations of the need for domain expertise is formula 409. Have either of you heard of Formula 409?

Justin Bui:

The cleaning solution?

Robert J. Marks:

The cleaning solution. Okay. I asked Daniel who is from Columbia and I ask 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 409 experiments in order to design that final result. And that required domain expertise. I'm sure it was done by chemist. I'm sure it wasn't done by junior high students, for example. In fact, if 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 a hundred 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. This was done by an industrial chemist. I think his name was Larsen. 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 contingencies, in order to do good design, if you will, we need this expertise. And that's something which is really, really important. Justin, you have any thoughts on this?

Justin Bui:

Yeah, kind of like we said, the addition of subject matter expertise really does reduce the complexity of things. I think my research tied a 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 stuff like that. 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 that's where a lot of the subject matter expertise is actually best used. Right? 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. Right? 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.

Robert J. Marks:

Sam, there are some other obstacles facing development of complex AI. We talked about, for example, in Jeopardy, of Watson repeating an incorrect answer. 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 unknowns 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. 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. 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, right?

Sam Haug:

That's right.

Robert J. Marks:

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. 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 is watching the contest, would give themselves a face palm 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 face palm 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 the main 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 counter argument. 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 out?

Sam Haug:

Yes. 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 it? A deep-

Sam Haug:

Deep level of-

Robert J. Marks:

... knowledge of domain expertise.

Sam Haug:

Yes.

Robert J. Marks:

That's a great.

Sam Haug:

Yes.

Robert J. Marks:

That's funny. Okay.

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 they're 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 exist 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. 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 a human being to withstand some things for trade 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 sin 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:

Well, 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 worked 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. We had one, and I remember it because it's so hilarious, it was a place called Charlie's Fish Market. At the time, there was an explosion in price of meat like pork and beef. Anyway, here 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.

Robert J. Marks:

Now, what do we mean by that? In our world now, there's a trade-off in performance. 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 a safe car, drive around in a Humvee, that's has extra armor plating on it. And if you want to grow down cheap, get a little scooter and don't wear a helmet or something.

Robert J. Marks:

But you have this entire gambit. And it turns out that Pareto trade-off says that 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, 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 experience 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.

Robert J. Marks:

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. 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'll perform well there," or, "I know that this particular aspect of the environment will cause to perform poorly." And so that can reduce the number of tests that have to be physically performed because the designer has enough demand 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, 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.

Robert J. Marks:

Okay.

Justin Bui:

The horse probably would've 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 of the hard tops. It turns out that they decided to replace all of the hard tops that were built or previously issued up to, I believe it was August. And when you observe 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 that, you're 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. Right? We talk about the known knowns, the known unknowns, and the unknown unknowns being kind of major hurdles. Right? And the unknown unknowns are the most dangerous kind because we don't know that we don't know them. 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.

Robert J. Marks:

Ah.

Justin Bui:

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. Right? 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?" Right? It's such a complex system with so many different subject expertise requirements that 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 overcomeable. Is that a word, overcomeable? 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 PhD students, Samuel Haug and Dr. Justin Bui, about obstacles of designing AGI. So until 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.