[SQUEAKING]

[PAPERS RUSTLING]

[CLICKING]

JEREMY KEPNER: All right. Welcome. Great to see everyone here. We're really excited about this opportunity. As you know, our AI accelerator has officially kicked off. All of your teams are ready to go. And we wanted this to be an opportunity as a team, come together and develop some common foundation, some common technological foundation, some common language for talking about these very challenging AI problems.

And so with that, I'll hand it over to Vijay.

VIJAY GADEPALLY: All right.

JEREMY KEPNER: Who will kick off with the first lecture, which basically provides some overview AI context for this.

VIJAY GADEPALLY: Again, welcome to the class. We're really looking forward to this. What we're going to present this morning is really a lot of overview material, right? Many of you here know a lot in AI and machine learning. This is really meant to just level set before we start the program, before we start these classes.

So you can see this generic title-- Artificial Intelligence and Machine Learning. And we're going to try and cover all of that in about an hour. So some details might be skipped, but we'll try and hit some of the salient features. All of these slides are available for you to use.

So if you're presenting back to your own teams, please feel free to pull from these slides.

We've actually gone through-- over some time putting a good set of survey and overview slides together. So if any of these are useful to you, just email us, or we'll make them available to you. You're more than welcome to use any and all of these slides if you're trying to present this back to other people.

So with that, let's begin. So we're going to do a quick overview of artificial intelligence.

Again, a lot of level setting going on here. We're going to do a quick, deep dive. These aren't the deepest of dives, again, given the amount of time that we have, but just talk very quickly about

supervised, unsupervised, and reinforcement learning, and then summarize.

And we can certainly stop for questions, philosophical debates, et cetera, towards the end. We'll try not to get a lot of the philosophical debates on camera if we can. All right. So first question-- what is artificial intelligence? And this is a question that probably a lot of you get. And I certainly have received this from a number of people. And that actually takes a lot of-- it took us a lot of time to come up with this.

And so we are very fortunate to have Professor Winston spend some time with us out at Lincoln Laboratory. And we actually brainstormed for a good hour or two, really trying to come up with what is a good definition for what we call artificial intelligence. And what we came up with is that there are two aspects to artificial intelligence.

First, that we should not confuse with each other. One is the concept of narrow AI, and another is a concept of general AI. And sometimes in conversation, we tend to conflate or mix the two. So narrow AI, according to our definition, is the theory and development of computer systems that perform tasks that augment for human intelligence, such as perceiving, classifying, learning, abstracting, reasoning, and/or acting.

Certainly in a lot of the programs that we work in, we're very focused on narrow AI and not necessarily the more general AI, which we define as full autonomy. So that's a very high-level definition of what we mean by AI. Now, many of you in the crowd are probably saying, well, AI has been around for a while. People have been talking about this for 50, 60-plus years. Why now? What is so special about it now? Why is this conversation piece now?

Well, from what we've seen, it really is the convergence of three different communities that have come together. The first is the community on big data. The second is a community on computing in a lot of computing technologies. And finally, a lot of research and results in machine learning algorithms.

The other one I forgot to put is dollar signs down here. People have basically figured out how to make money off of selling advertisements, labeling cat pictures, et cetera. So that's maybe the hidden-- why now in particular. But these are the three large technical areas that have evolved over the past decade or so to really make AI something we discuss a lot today.

So when we talk about AI, there are a number of different pieces which make up an AI system.

And we love the algorithms, people, but there is a lot more going on outside of that. So we've spent a

significant amount of effort just trying to figure out what goes into an AI system.

And this is what we call a canonical architecture. Very much in line with Lincoln Laboratory thinking, we like to think of an end to end pipeline. What are the various components?

And what are the interconnections between these various components? So within our AI canonical architecture shown here, we go all the way from sensors to the end user or missions.

And a lot of the projects that you all are working on are going to go all the way from here to there. A lot of our class, however, for the next few weeks is going to focus on step one, where a lot of people get stuck. So we take data that comes in through either structured or unstructured sources.

These are typically passed into some data conditioning or data curation step. This data is through that process, typically converted into some form of information. That information is then passed into a series of algorithms, maybe one or many algorithms. There are lots of them. There is life beyond neural networks.

Once we pass them through the algorithms, these typically form. This information is converted into knowledge. It's typically then passed into some module that interacts with the end user, or a human, or the mission. And that's what we call a human machine teaming step.

And that finally-- that knowledge with the human complement becomes insight that can then be used to execute the mission that the AI system was created for. All of these components sit on the bedrock of modern computing. Many different technologies that make up modern computing and the system that we're using today has combination of some of these computing hardware elements.

And certainly within the context of a lot of the projects that we are interested in, all of this also needs to be wrapped in a layer that we call robust AI, which consists of explainable artificial intelligence, metrics, and biased assessment, verification, validation, security, policy, ethics, safety, and training. We'll talk very briefly about each of these pieces in detail in a little bit.

As I mentioned, AI has an extremely rich history. This is just a very Lincoln and MIT specific view of the history of artificial intelligence. But certainly, there has been great work since the folks of Minsky, Clark, Dineen, Oliver Selfridge, et cetera, since the '50s. We've seen a lot of work in the '80s and '90s. And certainly, recently there has been, again, a resurgence of AI in our parlance in our thinking of the way AI works.

So without going into too much detail about each of these eras and why the winters came about, et

cetera, I think John Launchbury at DARPA actually put it very well, when he talked about different waves of AI technology that have come about. And when he talks about it, he talks about the three waves of AI or the four waves of AI. And the first wave, which you can think of as the first decade of AI technology, resulted in a lot of reasoning-based systems, which were based on handcrafted knowledge.

So an example of an output of this would be an expert system, right? So a lot of work in that. So if we take the four dimensions that John Launchbury suggests of the ability of the system to perceive, learn, abstract, and reason, these are typically pretty good at reasoning. Because they encoded human knowledge, right?

So a human expert sat down and said, what's going on in the system? And tried to write a series of rules. So tax software, for example, does a pretty reasonable job of that where a chartered accountant or a tax expert sits down, encodes a series of rules. We have a question in the back.

AUDIENCE: Yeah. [INAUDIBLE] I just wanted an example of an expert system from the '50s to [INAUDIBLE].

VIJAY GADEPALLY: Yep. So the question is, are there examples of expert systems? And certainly, one would be tax software. My graduate research was actually an autonomous vehicle.

Some of the early autonomous vehicles used a form of expert systems where the states on a finite state machine were maybe handcrafted.

And the transitions between them were designed that way. There was some machine learning wrapped around, but expert systems certainly played a large part in some of the early, even autonomous vehicle research that went on. All right. So over time, we were able to use these expert systems.

And don't get me wrong. These systems are still extremely valid in cases where you have limited data availability, limited compute power, a lot of expert systems still being used, or in cases where explainability is a very important factor. You still see expert systems, because they do have the ability to explain why they came up. They can typically point to a set of rules that somebody wrote, which is usually quite interpretable by a human.

However, as we were able to collect more data, we were maybe able to understand a little bit more about what was the underlying process. We were able to apply statistical learning.

And this led to the next era or next wave of AI technologies, which is often called the learning wave.

And this was really enabled by lots of data-enabled non-expert systems. So what we mean by that is we were able to dial back the amount of expert knowledge that we encoded into the algorithm, maybe put a higher level of expert knowledge into that. But usually, then used data to learn what some of these rules could be.

An example of that, in case someone wants to ask, would be in speech processing, for example. So we were able to say, well, I realized that speech follows this Gaussian mixture model. So I can encode that level of statistical knowledge, but I'm going to let the system figure out the details of how all that actually works out.

And there are many other cases. Again, coming back to some of the research I did on autonomous vehicles, we were able to maybe use some high-level expert rules, that here are a set of states that a car may be in. But I'm going to let the algorithm actually figure out when these transitions occur and what constitutes a transition between different states.

So looking at the four vectors that you could think about it, these systems had a little bit more on perception. Obviously, we're doing a lot more learning. But their ability to abstract and reason was still pretty low. And by reasoning, we mean, can you explain?

Can you tell us what's going on when you give me an output to the result?

The next wave which we're maybe at the beginning stages of is what we call contextual learning or contextual adaptation. This is where an AI system can actually add context into what it's doing. I'm not sure I have too many examples of people doing this very well.

I think most of the work today probably falls into the end stage of the learning wave of AI. But we're able to combine a bunch of these learning things to make it look like it's contextual in nature. But the key concept over here is being able to have the system automatically abstract and reason. So the way that we think about things, right?

So if I see a chair over here and I put a chair somewhere else, I still know it's a chair, because I'm using other contexts. Maybe it's next to a table or stuff like that. Some early research going on in that area. And certainly, the next wave of this is what we call abstraction. And there is very little work on this, but if we have to think out in the future.

But this is really the system of the ability of an AI system to actually abstract information that it's

learning. So instead of learning that a chair or a table is something with a leg at the bottom, it learns that a table is something you put things on and is able to abstract that information or that knowledge to any other domain or any other field. Do we have any questions before I continue from here? OK, great.

So that's a little bit on the evolution of AI. The reason we like to go through this is because there is great work going on in each of these waves. And nothing that people are doing in any of these waves is any lesser or more. It's typically dependent on what you have at your disposal.

What I like to tell people sometimes is the way to think about all of this is you have a couple of dials at your disposal-- turning dials. The first dial is, how much compute?

Right? How much ability do you have to crunch data? The second piece is, how much data do you actually have available? This can be labeled data in many cases.

And the third dial is, how much knowledge are you able to embed into an algorithm? In certain cases where maybe you have very little computing, very little labeled data availability, but a lot of expert knowledge or a lot of ability to encode information into an algorithm, you might be able to use an expert system, right? And that's a very good use case for that.

An example may be on another dimension where you want to encode very little human knowledge, but you have a lot of computing and data available, would be rare neural networks fall in, where they're essentially learning what the human knowledge-- what that encoded information should be.

A lot of statistical techniques also fall into that camp where maybe you encode a little bit of information as to what the background distribution of the process is. But it learns the details of exactly how that distribution is modeled, based on the data that it sees.

So you have a lot of different settings that you can use. And there are a number of different techniques within the broader AI context that you can use to achieve your mission. And I'm sure many of you are going to be doing different types of algorithms. And a lot of that decision will be dependent on, well, how much data was I given? Right? How good is this data that I'm using? Is there an ability to learn anything from this?

And if not, you might have to encode some knowledge of your own into it saying, well, I know that this process looks like that, so let me tell the algorithm to not waste too much time crunching the data to learn the underlying distribution, which I can tell you. Why don't you learn the parameters of the

distribution instead? Does that makes sense? All right.

And as you know, there's just a lot going on in AI and machine learning. You can't walk two steps without running into somebody who's either starting something up, working for one of these organizations. So it really is an exciting time to be in the field. All right.

So that's a little bit on the overview, but let's talk now in a little bit of detail on what some of the critical components are within this AI architecture. So one thing that we like to note-- and there's a reason that as we've been reaching out to a number of you, we've been talking about getting data, right? Work with stakeholders to get your data in place.

And the reason we talk about that is data is critical to breakthroughs in AI. A lot of the press may be on the algorithms and the algorithms that have been designed. But really, when we've looked back in history, we've seen that, well, the availability of a good canonical data set actually is equally, if not more critical to a breakthrough in AI.

So what we've done here is we've just picked a select number of breakthroughs in AI. Our definition of a breakthrough in this particular example is something that made a lot of press or something that we thought was really cool. So here are some examples of that in different years.

And we've talked about the data set, the canonical data set that maybe led to that breakthrough or that was cited in that breakthrough as well as the algorithms that were used in that breakthrough and when they were first proposed. This is notional in nature. Clearly, you could adjust these dates a few years here or there.

But what we really want to get across is that the average number of years to a breakthrough from the availability of a cool data set or a very important, well-structured, well-labeled data set is much smaller than from the algorithm's first proposal or when the algorithm first comes out.

So as you're developing your challenge problems, as you're developing your interactions with stakeholders, certainly something to keep in mind that there's clearly a lot of algorithmic research that's going to go on. But having a good, strong, well-labeled, and documented data set can be equally important. And making that available to the wider AI community, the wider AI ecosystem can be very, very valuable to your work and the work of many other people.

All right.

So back to the AI architecture, we're going to just go through very briefly-- different pieces of this

architecture. So the first piece we're going to talk about is data conditioning, which is converting unstructured and structured data. Within the structured and unstructured data, you might have structured sources coming from sensors, network logs for some of you, metadata associated with sensors, maybe speech or other such signals.

There's also a lot of unstructured data. You think of things that you might collect from the internet that you might download from, say, a social media site, maybe reports, other types of sensors that maybe don't have well-- that don't have the strong structure within the data set itself.

And typically, this first step, what we call the data conditioning step, consists of a number of different elements. You might want to first figure out where to put this data.

That can often take a lot of time. And there have been religious wars fought on this topic.

We're here to tell you that you're probably OK, picking most technologies. But if you have any questions, feel free to reach out to me or to others on the team. We have a lot of opinions on what's the right infrastructure to solve the problem. Typically, these infrastructure or databases might provide capabilities such as indexing, organization, and structure.

Very important in unstructured data to convert it into some format that you can do things with.

They may allow you to connect to them using domain specific languages. So it's converting it into a language that maybe you're used to talking. They can provide high-performance data access and in many cases, a declarative interface. Because maybe you don't really care about how the data is being accessed. You want to just say select the data, give it to me. And then move forward from there.

Another important part of the data conditioning step is data curation. This unfortunately will probably take you a very long time. And it requires a lot of knowledge of the data itself, what you want to do with the data, and how you receive the data.

But what you might do in the data curation step is perform some unsupervised learning, maybe reduce the dimensionality of your problem. You might do some clustering or pattern recognition to maybe remove certain pieces of your data or to highlight certain pieces of the data that look important.

You might do some outlier detection. You might highlight missing values. There's just dot, dot, dot, et cetera, et cetera, et cetera. A lot goes on in the data curation step. We could certainly spend hours

just talking about that. And the final thing, especially within the context of supervised machine learning, but even in the world of unsupervised learning would be spending some time on data labeling, right?

So this is taking data that you've received, typically doing an initial data exploration.

Could be as simple as opening it up in Excel to see what the different columns and rows look like if that's a suitable place to open it up. You might look for highlight, missing, or incomplete data, just from your initial data exploration.

You might be able to go back to the data provider or to the sensor and say, can you reorient the sensors or recapture the data? I noticed that every time you've measured this particular quantity, it always shows up as 3. I can't imagine that that's correct. Can you go back and tell me if that sensor is actually working? Or is it actually 3? In which case, you might want to know that.

And you might look for errors, biases, and collection, of course, on top of the actual labeling process that you're doing to highlight phenomenology within the data that you'd like to then look for through your machine learning algorithms. I'll pause for a second. Yes?

AUDIENCE: I have a quick question.

VIJAY GADEPALLY: Yeah?

AUDIENCE: What's the ratio that you see between structured data and unstructured data?

VIJAY GADEPALLY: So the question is, what's the ratio we see between structured and unstructured data? That's a great question. So the ratio in terms of the volume or the ratio in terms of what you can do with it? Because those are actually almost the opposite.

So, again, I'm talking about a few data sets that I'm very familiar with. The unstructured data can often be 90% of the volume. And maybe the 10% is the metadata associated with the unstructured data. Most of the value, however, comes from the structured data where people really analyze the crap out of these structured data, because they know how to.

There is certainly a lot of potential within the unstructured data. So when we talk to people, that's why we talk a lot about infrastructure and databases as being an important first step. Because if you can just take the unstructured data and put it into a structured or semi-structured form, that itself can provide a lot of value.

Because very often in problems that we see, the 90% volume of data is largely untapped.

Because people don't know how to get into it, or don't know what to do with it, or it's not in a form that you can really deal with. So I think next class, we're going to be talking to you about how to organize your data, strategies for organizing data that can get you a lot more value out of the unstructured data. Does that answer your question? Yes?

AUDIENCE: [INAUDIBLE]

VIJAY GADEPALLY: So the question is when you apply AI or machine learning techniques to a problem domain, is it typically a single modality or multiple modalities? I'd say the answer is both. Certainly, there's a lot of research. And back there, we have Matthew, who's actually doing research on that right now on how to fuse multiple modalities of data.

I know a lot of projects that are being discussed here are certainly looking at multiple modalities.

If I had to say as of today, a lot of the work that's out there-- the published work may be focused on a single modality. But that's not to say-- I mean, I think there is a lot of value on multiple modalities. But the challenge still comes up on, how do you integrate this data, especially if they're collected from different systems? Yep?

AUDIENCE: Just on the structure versus unstructured. So it's not really my area, but I am surprised to see speech in the structured [INAUDIBLE]. And I wonder. Is that just because the technologies that can force the [INAUDIBLE] and all of this data conditioning are mature enough that you can basically treat it [INAUDIBLE]?

VIJAY GADEPALLY: So the question is, why would speech or something else like that fall into structured versus unstructured? And you're absolutely right. I think when we pick speech-- and I'm sure there are others in the room that might disagree with that and might stick it over here.

When we look at the type of acquisition processes that are used, the software that's used, they typically come out with some known metadata. They follow a certain pattern that we can then use, right? There is a clear range to where there is-- the frequency to which the data is collected. And that's why we stuck it in the structured data type.

Of course, if you're collecting data out in the field without that, you could probably stick it into the unstructured world as well. But that's probably a good example of something that can fall in between the two places. OK. All right. Now, for the part everyone's really interested in-- machine learning,

right?

So, all right, you got through the boring data conditioning stuff, which will take you a couple of years or something like that. Nothing serious. And now, you're ready to do the machine learning. And now, you're given a choice. Well, which algorithm do you use?

Neural networks, you might say, right?

There is a lot more, though, beyond the neural network world. So there is numerous taxonomies.

I'm going to give you two of them today for how you describe machine learning algorithms.

One that's really an interesting way is from Pedro Domingos at the University of Washington in which he says that there are five tribes of machine learning.

So there are the symbolists, which an example of that would be expert systems. There are the Bayesian tribes, which an example of an algorithm within that might be naive Bayes.

There are the analogizers, which an example of that would be a support vector machine and the connectionists, an example of which would be deep neural networks. And evolutionary is an example of that which might be genetic programming.

What really I'm trying to get across-- I'm sure the author is trying to get across here is that lots and lots of different algorithms. Each have their relative merits and relative strengths. Apply the right one for your application. Apply the right one for-- again, given these dials that I talked about earlier, the amount of computing that you have available, the amount of data that you have available, and the amount of expert knowledge that you're able to encode into your algorithm that you think is generalizable enough.

If we actually talk about-- this is a very useful chart I found in describing to folks that are not familiar with AI that might say, wasn't AI just neural networks? And neural networks are a part of AI, but not necessarily all of it. So if we think of the big circle is the broad field of artificial intelligence, within that is the world of machine learning.

Within machine learning are connectionists or neural networks that fall into a small camp within that. And deep neural networks is a part of neural network. So can anyone maybe give me an example-- although, I've said it numerous times-- of something that might fall out of machine learning, but into artificial intelligence from an algorithmic point of view? Yes?

AUDIENCE: Graph search.

VIJAY GADEPALLY: Graph search could be an example. I would maybe stick that into some of the connectionists, however.

AUDIENCE: Expert systems.

VIJAY GADEPALLY: Yes, exactly. So expert systems-- it's the one that comes to my mind. Or knowledge-based systems are an example of maybe something that fall outside of the realm of machine learning, again in the very strict sense, but maybe within the realm of artificial intelligence from an algorithmic point of view.

OK, so that's a little bit on the algorithms. Next, let's talk about some of the modern computing engines that are out there. I mentioned that data compute as well as algorithms have been key drivers to the resurgence of AI over the past few years. What are some of these computing technologies, for example? So clearly, CPUs and GPUs. They're very popular computing platforms. Lots of software written to work with these computing platforms.

But what we're seeing now is that with the end of Moore's Law and a lot more performance engineering going on, we're seeing a lot more work, research, and hardware architectures that are custom in nature. And custom architectures are almost the new commercial off-the-shelf solutions that are out there.

So an example of a custom architecture could be Google's Tensor Processing Unit, or TPU.

There is some very exciting research going on in the world of neuromorphic computing.

I'm happy to chat with you all later if you're interested to know what's going on in that area and maybe our role in some of that work.

And there is just some stuff that we would still call custom. These are still people deciding, designing, basically looking at an algorithm saying, OK, here's the data layout.

Here is the movement of data or information within this algorithm. Let's create a custom processor that does that. An example of that could be the graph processor, which is being developed at Lincoln Laboratory.

And obviously, no slide on computing architectures or computing technologies would be complete without mentioning the word quantum in it. There is some early results on solving linear system of

equations. But I think applied to AI, it's still unsure, or unknown, or unproven where quantum may play a part. But certainly, a technology that all of us, I'm sure, have heard of, or continue to track, or just interested in seeing where that goes to.

So within the first few, however, these are all products that you can buy today. You can go out to your favorite-- your computing store and just purchase these off-the-shelf solutions.

A lot of software has been written to work with these different technologies. And it's a really nice time to be involved. Yeah?

AUDIENCE: Can you give a brief just concept of what is attached to the [INAUDIBLE]? I see [INAUDIBLE], but I don't really have a high-level concept of why I should associate with that.

VIJAY GADEPALLY: OK, so the question is, what should I think about when I'm thinking about neuromorphic? So there's a few features which I say fall into the camp of what people are calling neuromorphic computing. One is what they're calling a brain inspired architecture, which often means its clockless.

So you typically have some-- so a lot of these technologies have clocked movement of information.

These might be clockless in nature. They typically sit on top of different types of memory architectures.

And I'm trying to think of what would be another parameter that would be very useful. I can probably send you a couple things that help highlight that. I certainly wouldn't call myself an expert in this area.

AUDIENCE: OK, thanks.

VIJAY GADEPALLY: But, yeah. I think the term that's used is it's supposed to mimic the brain in the way that the computing architecture actually performs or functions. So lots of research as well. And this is work that we've done here at the lab on actually trying to map the performance of these different processors and how they perform for different types of functions.

So what we're doing here is basically looking at the power on the x-axis. And the y-axis is the peak performance in giga operations per second. Different types of precision are noted over there by the different shapes of the boxes and then different form factors.

And the idea here is basically to say there's so much going on in the world of computing.

How can we compare them? They all have their own individual areas where they're strong.

So one can't come up and say, well, the GPU is better than the CPU. Well, it depends on what you're trying to do and what your goals of the operation are.

So some of the key lines to note here is that there seems to be a lot of existing systems on this 100 giga operations per watt on this line over here, this dash line. Some of the newer offerings maybe fit into the 1 tera op per watt. And some of the research chips like IBM's TrueNorth or Intel's Arria fall into just a bit under the 10 tera operations per watt line that we see there.

But depending on the type of application, you may be OK with a certain amount of peak power. So if you're looking at embedded applications, you're probably somewhere over here, right?

If you're trying to get something that's on a little drone or something like that, you might want to go here. And if you have a data center, you're probably OK with that type of power utilization or peak power utilization. But you do need the performance that goes along with that.

So I'd say the most important parts to look at are essentially these different lines.

Those are the trajectories for maybe some of the existing systems, all the way up to some of the more research oriented processors out there. OK? All right.

So we talked about modern computing. Let's talk a little bit about the robust AI side of things. And the basic idea between robust AI is that it's extremely important. And the reason that it's important is that the consequence of actions on certain applications of AI can be quite high.

So what we've done here is think about, where are the places that maybe humans and machines have their relative strength? So on the x-axis, we're talking about the consequence of action.

So this could be, does somebody get hurt if the system doesn't do the right thing? All the way down to, no worries if the system doesn't do the right thing, which could be maybe some of the labeling of images that we see online, might fall into this category.

I'm sure people disagree with me on that.

But maybe a lot of national security applications. Health applications certainly fall into the area of high consequence of action. If you give someone the wrong treatment, that's a deal. And then on the y-axis, we're talking about the confidence level in the machine making the decision. So how much

confidence do we have in the system that's actually making the decision?

In certain cases, we might have very high confidence in the system that's making a decision.

And obviously in certain cases, we do not have much confidence in the system making the decision. So in areas where you have a low consequence of action, maybe high confidence level in the machine making the decision, we might say those are best matched to machines.

Those are good candidates for automation.

On the contrary, there might be areas where that consequence of action is very high. And we have very little confidence in the system that's making the decision, probably an area we want humans to be intricately, if not solely or involved or responsible. And the area in between is where machines might be augmenting humans.

Does anybody want to venture maybe a couple of examples-- help come up with a couple of examples here that we might put into each of these categories? So maybe what's a good problem that you can think of that might be best matched to machines, beyond labeling images for advertisements?

AUDIENCE: Assembly lines.

VIJAY GADEPALLY: Assembly lines? Yep, that's a good example. I'm thinking within spam filtering, could be another example where-- I mean, there is some machine augmenting human. It does send you an email saying this is spam. Are you sure? But for the most part, it's largely automated.

I'd say a lot of the work that many of us are probably doing falls into this category, maybe on different sides of the spectrum, but of where machines are augmenting humans.

So the system can be providing data back to a human that can then select. It might filter information out for humans that then the humans can then go ahead and say, OK, well, instead of looking at a thousand documents, I can only look at 10, which is much better.

And then there's obviously certain-- probably we want humans to be heavily involved with any kinetic-- anything that involves life or death-- we probably want. And there are probably legal reasons, also, that we want humans involved with things like that.

One of the examples that we often get which is autonomous vehicles-- and it's always a little confusing where autonomous vehicles fall into this. Certainly, the consequence of action of a mistake

in an autonomous vehicle can be pretty high.

And as of today, the confidence and the decision-making is medium at best. But people still seem to somehow be OK with fully automating. That just shows how terrible Boston roads or driving in general is, that we're like, I'm not really sure if this thing will kill me or not, but totally worth trying it out.

AUDIENCE: Do you think the trend in this chart is to slowly expand the yellow out?

VIJAY GADEPALLY: Yes, I'd say-- the question is, is the yellow expanding? I think so. One could make the argument that, is it shifting that direction? Are we finding areas where-- and I think that's maybe the direction. We are probably looking at automating certain things a little bit more as confidence in decision-making goes up.

So you might think about this frontier moving down so that maybe the green expanding slightly and the yellow taking over a little bit of the red. There might be some places where over time, we're more open to the machine making a decision and the human having a largely supervisory role, which I would put right at this frontier between the yellow and the red.

AUDIENCE: Again, I guess it depends on what augmenting means. But I guess [INAUDIBLE]

is truly red without any-- even cognitive augmenting.

VIJAY GADEPALLY: I can think of some examples, but maybe I'll share it with you later. So certainly, a robust artificial intelligence plays a very important part in the development and deployment of AI systems. I won't go through the details of each of these.

I'm sure many of you are very familiar with it. And I know a few of you are far more knowledgeable about this, maybe than I am. But some of the key features would be explainable AI, which is a system being able to describe what it's doing in an interpretable fashion.

Metrics-- so being able to provide the right metric if you want to go beyond accuracy or performance. Validation and verification-- there might be cases where you're not really concerned about the explainability, but you just want to know that when I pass an input, I get a known output out of it. And is there a way to confirm that I'm able to do that?

Another could be on security. So an example of this-- or not having security would be counter AI, right? So when we talk about security within the context of robust AI, it's almost like the cryptographic way of thinking about it, which is, can I protect the confidentiality, integrity, and

availability of my algorithm, the data sets, the outputs, the weights, the biases, et cetera?

And finally, a lot of significant importance is policy, ethics, safety, and training. This is actually very important in some of those applications where in the previous slide, we had the yellow and the red where humans and machines augmenting humans, where that falls.

A lot of that might be governed by policy, ethics, safety, and training, which is some of the examples that I can think of, where there are policy reasons that make it that only a human can be involved with this, maybe with minimal input from a system. OK.

And the final component of our AI architecture-- we've gone through conditioning, algorithms, computing, robust AI-- is human machine teaming. And I think what we want to get across with human machine teaming-- that is it really depends on the application and what you're trying to do. But it is important to think about the human and the machine working together.

And there is a spectrum of where the machine will largely-- will play a large part and the human largely supervisory or to where the human plays a large part and the machine is very targeted in what you do with the machine or the AI of the system.

But a couple of ways to think about it would be-- of course, we talked about the confidence level versus consequence of actions, but also, the scale versus application complexity. So on the top chart over there, we have on the x-axis is the application complexities. How complex is this application? And on the y-axis is the scale. How many times do you need to keep doing this thing?

Places that machines might be more effective than humans are where we have low application complexity, but very, very high skill. So again, spam filtering falls into this. The complexity of spam filtering has gone up over time, but is something that is reasonable within systems. But the scale is very high, that we just don't want a human being involved with that process.

And on the other end of the spectrum is where you have very high application complexity that'll only happen a couple of times. So this could be, say, reviewing a situation.

Maybe a company is trying to make an acquisition. It's not going to happen over and over.

So you might have a human involved with that, that goes through a lot of that. Maybe they target the system to go look for specific pieces of information. But really, it's the human that might be more effective in that, especially given that the situation would change over and over. All right.

So with that, we're going do take a quick tour of the world of machine learning. I'll stop there for a second. Any questions? OK. All right. So what is machine learning? Always a good place to start. It's the study of algorithms to improve their performance at some task with experience. In this context, experience is data.

And they typically do this by optimizing based on some performance criteria that uses example data or past experience. So in the world of supervised learning, that could be the example data. Or past experience could be the correct label, given an input data set or input data point.

Machine learning is a combination of techniques from statistics, computer, and computer science communities. And it's the idea of getting computers to program themselves. Common tasks within the world of machine learning could be things like classification, regression, prediction, clustering, et cetera.

For those who are maybe making the shift to machine learning from traditional programming, I found this, again, from Pedro Domingos to be a very useful way of describing it to people.

So in traditional programming, you have a data set. You write a program, which would be if you see this, do that. When you see this, do that. For this many instances, do the following thing on it and then write an output out, right?

So you input a data into the program into a computer, and the computer produces an output where it says, OK, I've applied this program on that data. And this gives me the output.

Machine learning is a very different way of thinking about it in which you're almost inputting the data as well as the output.

So in this case, the data could be unlabeled images. The output could be the labels associated with those images. And you tell the computer, figure out what the program would look like.

And this is a slightly different way of thinking about machine learning versus traditional programming.

What are some of these programs or algorithms that the computer might use to figure it out?

So within the large realm of machine learning, we have supervised, unsupervised reinforcement learning. What we have in the brackets is essentially what you're providing in the world.

In the case of supervised learning, you're providing labels, which is the correct label associated with

an input feature or with an input data set or data point. In unsupervised learning, you typically have no labels, but also are limited by what the algorithm itself can do.

And in the world of reinforcement learning, instead of a label per data point, you're providing the reward information to the system that says if you're doing more-- if you're doing the right thing, I'm going to give you some points. If you're doing the wrong thing, I'm going to take away some points-- very useful in very complex applications where you can't really figure out the labels associated with each data point.

Within the world of supervised learning, the typical tasks that people have-- and I should note before I go through this. There's a lot of overlap between all of these different pieces. So this is a high-level view. But we can certainly argue about the specific positioning of everything. I'm sure we can.

So within supervised learning, you can fall into classification regression. Unsupervised learning is typically clustering dimensionality reduction. And within these, there are different algorithms that fall into place. So examples could be things like neural nets that cover all of these spaces.

You got-- just take regression, PCA, which might fall into dimensionality reduction, lots and lots of different techniques and also some in the reinforcement learning world.

And there's just more and more and more. If you open up a survey of machine learning, it'll give you even more than all of these techniques over here.

And the thing to remember when you're using machine learning is that there are some common pitfalls that you can fall into. An example of that would be overfitting versus underfitting where you come up with this awesome model that does really, really well on your training data.

You apply it to your test data, and you get terrible results. You might have done a really good job learning the training data, but not necessarily learning-- being able to generalize beyond that. Sometimes it could be just the algorithm itself is unable to correctly model the behavior that's exhibited by the training and test data.

I won't go through each of these again, but there might just be bad, noisy, missing data.

That certainly happens where you end up with an algorithm with terrible results. And you look at it and you're like, well, why is that? And you actually look at the data that you did. And it was incorrect, that there was just missing features. Or it was noisy in nature, such that the actual phenomenology that you were trying to look for was hidden within the noise.

You might have picked the wrong model. You might have used a linear model in a non-linear case where the phenomenology you're trying to describe is non-linear in nature, but maybe you've used a linear model. You've not done a good job of separating training versus testing data, et cetera, et cetera.

So we'll just take a quick view into each of these different learning paradigms. So the first is on supervised learning. And you basically start with label data or what we call-- it was often referred to as ground truth. And you build a model that predicts labels, given new pieces of data.

And you have two general goals. One is in regression, which is to predict some continuous variable or a classification, which is to predict a class or label. So if we look at this, the diagram on the right, we have training data that we provide, which is data and labels.

That goes into a trained model. That's typically an iterative process where we find out, well, did we do a good job? That is now called a supervised learning model that we then apply new data, or test data, or unseen data and look at the predicted labels.

Typically, when you are designing an algorithm like this, you'd separate out. You'd take your training data. You'd remove a small portion of it that you do know the labels for. That's your test data over here. And then you run that. And you can see, well, is it working well or not?

And most of these algorithms have a training step that forms a model. So when we talk about machine learning in both the supervised and unsupervised sense, we'll often talk about training the model, which is this process, and then inference, which is the second step, which is where you apply unseen data. So this is the trained model in deployment or in the field. It's performing inference at that point.

Of course, no class these days on machine learning and AI could go without talking about neural networks. And as I mentioned, neural networks do form a very important part of machine learning. And they certainly are an algorithm that many of you, I'm sure, are familiar with. And they fall well within the supervised and unsupervised. And they've been used for so many different applications at this point.

So what's a neural network? A computing system inspired by biological networks. And the system essentially learns by repetitive training to do tasks based on examples. Much of the work that we've seen is typically it being applied to supervised learning, though I'll mention some that we are doing, some research and actually applying it for unsupervised learning as well. And they're quite powerful.

The components of a neural network include inputs, layers, outputs, and weights. So these are often the terms that someone will use. And a deep neural network has lots of hidden layers. Does anyone here have a better definition for what deep neural network means beyond lots? I've heard definitions anywhere, 3 and above. Yes?

AUDIENCE: [INAUDIBLE] deep neural network will occur at any recurrent networks. Because that has more than one layer, but not necessarily more than one layer after you have actually written the code for it.

VIJAY GADEPALLY: OK, so one definition here for deep is-- and this is-- anyone have a better-- no. So the one to beat right now is-- a feature of a deep neural network could be recurrence within the network architecture, which implies that there is some depth to the overall network. So above 3 with recurrence-- deep. All right.

Lots of variance within the supervised world of neural networks, such as convolutional neural networks, recursive neural networks, deep belief networks. One, I think, in my opinion-- again, since you've all asked me to opine here. I know you've not, but I think a reason that these are so popular these days is there's so many tools out there that are very easy to use.

You can just go online and within about five minutes, write your first neural network.

Try writing a hidden mark-off model that quickly. Maybe there are people who can, but in general.

So what are the features of a deep neural network? So you have some input features.

You have weights, which are essentially associated with each line over here, as well as biases for each of the layers that govern the interaction between the layers and then an output layer.

So these input features can often be combined to each other. So these feature vectors that are coming in can often be combined. Think Jeremy we'll talk a little bit about how the matrix view of all of this.

But you can think of it as if your-- an example could be if you have an image, it could be the RGB pixel values of each pixel in that image, could be the input feature. So you could have large numbers of input features. If you have a time series signal, it could be the amplitude or the magnitude at a particular frequency or at a particular step.

There's often a combination of features that you might do. So in addition to the pixel intensities for

an image, you might also then combine the spatial distance between two pixels.

Or its position within the image may also be another input feature. And you can really go hog wild over here, just trying to come up with new features.

And there's a lot of research just in that area, which is I take a data set that everyone knows. And I'm just going to spend a lot of time doing feature engineering, which is coming up with, well, what is the right way to do the features? So coming back to an earlier question, this is an area where people are often looking at supplementing maybe a given data set with additional data.

And then fusing those two pieces together, for example, could be audio and text together as input features to a network, that you can then learn that might do a better job. But all of this is governed by this really, really simple, but powerful equation, which is that the output at the i plus 1th layer is given by some non-linear function of the weights multiplied by the inputs from the previous step, plus some bias term then.

And when you're learning-- when you're training a machine learning model, you're essentially trying to figure out what the Ws are and what the Bs are. That's really what a model is defined as. So if we zoom into one of these pieces, it's actually pretty straightforward what's going on over here.

So you have your inputs that are coming from the previous layers, so this could be your Y sub i. Here are the different weights, so W1, W2, W3. These are the connections or the weights going into a neuron or a node. And you're performing some function on these inputs.

And that function is referred to as an activation function.

So let's just take an example where we have some actual numbers. Maybe I've gone through.

I've trained my models. I figured out that just for this one dot in that big network that we saw earlier, that my weights are 2.7, 8.6, and 0.002. My inputs from the previous layer is maybe -0.06, 2.5, 1.4.

And all I'm doing is coming up with this x, which is -0.06 multiplied by 2.7, plus 2.5, times 8.6, plus 1.4 times that. That gives me some number-- 21.34. I apply my non-linear function, which in this case is a sigmoid governed by that equation at the top right.

And I say f of 21.34, so somewhere way over there is approximately 1, right? So this-- probably a little less than 1, but approximately 1 for the purpose of this.

And you just do that over and over. So really, a neural network-- I think the power of a neural network is it allows you to encode a lot less information than many of the other machine learning algorithms out there at the cost, typically, of a lot more data being used and a lot more computing being used. But for many people, that's perfectly fine, right?

But it does take-- it's just over and over, back and forth, back and forth, back and forth to come up with, what's the right Ws in order for this to give me a result that looks reasonable?

Lots of work going on and just deciding the right activation function.

I showed you a sigmoid over there. We do a lot of work with ReLU units. The choices-- there are certain applications-- certain, I should say, domains or applications where people have found that a particular activation function tends to work well.

But that choice is something I leave to domain experts to maybe look at their problem and figure out what are the relative advantages. Each of these have their own advantages. I know, for example, one of the big advantages of rectified linear unit is that since you're not limiting yourself between a -0 and 1 range, you don't have to do that. You don't run into a problem of vanishing gradients. That doesn't mean much for people. That's OK. We're not going to spend too much time talking about that anyhow.

AUDIENCE: Vijay?

VIJAY GADEPALLY: Yeah?

AUDIENCE: So in general, [INAUDIBLE].

VIJAY GADEPALLY: So the question is picking the activation functions, picking the number of layers. We'll talk about that in a couple of slides. But there is a lot of art. Trial and error-- yes, but also, we'll call it art as well that's involved with coming up with that.

A lot of what happens in practice, however, is you find an application area, which looks very similar to the problem that you are trying to solve. And you might borrow the architecture from there and use that as a starting point in coming up with where you start. Yeah?

AUDIENCE: Are you aware of any research of some type of parameterizing the activation function and then trying to learn the activation function?

VIJAY GADEPALLY: I'm sure people are doing it. I'm personally not familiar with that research. I don't if

anyone else in the room has-- yep?

AUDIENCE: [INAUDIBLE] DARPA D3M program, so data-driven machine learning. You're trying to learn both the architecture of the network and the activation function and therefore all the other attributes. Because you're trying to just go from data set to machine learning system with no human intervention.

VIJAY GADEPALLY: So the question was, is there any research into parameterizing the activation function? So I guess the model as a whole. So, yeah, there is. And one of the responses was that there is a program run by DARPA, which is the D3M program, which is really looking at, can you go from data to result with no or almost no human intervention?

I'm not familiar with activation function parameterization. But certainly, network model parameterization is absolutely there. So people who are running optimization models to basically look for-- I have this particular set of resources. What is the best model architecture that fits into that?

Maybe I want to deploy this on a really tiny processor that only gives me 16 megabytes of memory. I want to make sure that my model and data can fit on that. Can you find what would be the ideal model for that? So that's absolutely something that people are doing right now. But I'm not sure if people are trying to come up with, I guess, brand new activation functions. All right.

So lots of stuff in the neural network landscape. And as I mentioned earlier, neural network training is essentially adjusting weights until the function represented by the neural network essentially does what you would like it to do. And the key idea here is to iteratively adjust weights to reduce the error.

So what you do is you take some random instantiation of your neural network or maybe based on another domain or another problem. You might borrow that. And you start there. And then you pass a data set in. You look at the output and you say, that's not right. What went wrong over here?

And you go back and adjust things and do that again, and again, and again, and again, and again, over and over, until you get something that looks reasonable. That's really what's going on over there. And so real neural networks can have thousands of input, data points-- hundreds of layers, and millions to billions of weight changes per iteration. Yes?

AUDIENCE: So what you're talking about is [INAUDIBLE] adjustment [INAUDIBLE]. Do you know of any [INAUDIBLE] this process?

VIJAY GADEPALLY: Yes, there's a lot of work being done to parallelize this-- AUDIENCE: Like, for-- VIJAY GADEPALLY: --and by default.

AUDIENCE: [INAUDIBLE]?

VIJAY GADEPALLY: So the question is when-- as I just described it right now, it's a serial process where I pass one data point in. It goes all the way to the end. It says, oh, this is the output-- goes back and adjusts. Are there techniques that people are doing to do this in a distributed fashion? And the answer to that is a strong yes. It's a very active area in especially high-performance computing and machine learning.

We might talk about this in-- are we talking about this on day three? We might talk a little bit about it. But there is model parallelism, which is I have the model itself distributed across multiple pieces. And I want to adjust different pieces of the model at the same time. There's research and lots of results. I think we might even have some examples that people are doing with that.

AUDIENCE: Have you got some examples on the [INAUDIBLE] approach, the [INAUDIBLE] approach?

VIJAY GADEPALLY: A little bit earlier.

AUDIENCE: Communication [INAUDIBLE].

VIJAY GADEPALLY: So there are many different ways to parallelize it. One would be data parallelism, which is I take my big data set or big data point, and I distribute that across my different nodes. And each one independently learns a model that works well. And then I do some synchronization across these different pieces.

There are also techniques where you have-- the model itself may be too big to sit on a single node or a single processing element. And you might have to distribute that. So, yes, a lot of very interesting research going on in that area. And by default, when you do run things, they are running in parallel, just even on your GPU. They're using multiple cores at once. So there is some level of-- within the node itself, parallelism that runs by default on most machine learning software.

So inferences-- I mentioned is just using the trained model again. And the power of neural networks really falls within their non-linearity. So you have that non-linear F function that you're applying over and over and over across your layers. And this crudely drawn diagram on my iPad-- this is not clear at all.

If you had Xs and Os, right? It reminds me of a song. And you have features over here.

And you're trying to basically classify it. Which is an X? And which is an O? A linear classifier could do a pretty good job in this type of situation. And you could apply a neural network to this, but maybe it's overkill in that type of situation.

But in some feature space, if this is how your Xs and Os are divided amongst each other and you're trying to come up with the right label, one thing I might suggest is maybe find another feature space that you could maybe get a better separation between the two. Or a technique like a neural network might do a very good job. Or any of these non-linear machine learning techniques might do a very good job for looking for these really complex decision boundaries that are out there. All right.

So you mentioned earlier when you're designing a neural network, what do you have to do?

What are the different choices, et cetera? There is a lot going on here. So you have to pick the depth, the number of layers, the inputs, and what the inputs are, the type of network that you're using, the types of layers, the training algorithm and metrics that you're using to assess the performance of this neural network.

The good thing, however, is it so expensive to train a neural network, that you largely are not making these decisions in many cases. You just pick up what somebody else has done, and you start from there, and then you start. That might be-- I don't know if that's a good or a bad thing. But that's often a way in practice that people end up doing this.

But there is some theory on the general approach. I think in this short amount of time, which I'm already over, we won't be able to get into it. But I'm happy to-- actually, these slides have backups on them. So when I share them with you, they do have a lot more detail on each of these different pieces. All right.

Very quickly, we'll talk about unsupervised learning. And the basic idea is the task of describing a hidden structure from unlabeled data. So in contrast to supervised learning, we are not providing labels. We're just giving the algorithm a data set and saying, tell me something cool that's going on over here.

Now, clearly, you can't label the data if you do that. But what you can do is maybe look for clusters or look for dimensions or pieces of the data that are unimportant or extraneous. So if we observe certain features, we would like to observe the patterns amongst these features.

And the typical tasks that one would do in unsupervised learning is clustering and data projection, or data pre-processing, or dimensionality reduction. And the goal is to discover interesting things about the data set, such as subgroups, patterns, clusters, et cetera.

In unsupervised learning-- one of the difficulties in supervised learning-- we know, right? We have an input. We have a label. And we're like, OK, if that input-- if my algorithm doesn't give me the label, bad. Go retrain. Or I know what-- I can go back, use that as my performance metric.

On unsupervised learning, there is no simple goal such as maximizing a certain probability for the algorithm. Some of that is going to be something that you have to work on, is at the interclass or intraclass distance that I'm most having that separation. Is that going to be my performance metric? Is it the number of clusters that I'm creating? Is that the number of-- is that the metric that I'm using?

But it is very popular, because it works on unlabeled data. And I'm sure many of us work on data sets, which are just too large or too difficult to sit and label. An example that comes to my mind, certainly, is in the world of cybersecurity where you're collecting billions and billions of networked packets. And you're trying to look for an almost behavior.

You're not going to go through and look at each pack and be like, bad, good, what it is. But you might use an unsupervised technique to maybe extract out some of the relevant pieces, then use a supervised-- then go through the trouble of labeling that data, and then pass that on to a supervised learning technique. And I'm happy to share some research that we've been doing on that front.

Some common techniques are within clustering and data projection. Clustering is the basic idea that we want to group objects or sets of features, such that objects in the same cluster are more similar to those of another cluster. And what you typically do for that is you put your data in some feature space, and you try to maximize some intracluster measure, which is basically saying, I want the points within my cluster to be closer than anything outside of my cluster, right?

So that's a metric. And you iteratively move the membership from each. You set a number of clusters, saying, I need five clusters. It'll randomly assign things. And it'll keep adjusting the membership of a particular data point within a cluster, based on a metric such as the squared error.

So in this example, we might say that, OK, these are three clusters that I get out of it. Dimensionality reduction is the idea of reducing the number of random variables under consideration. Very often, you'll collect a data set that has hundreds to thousands of different features.

Maybe some of these features are not that important. Maybe they're unchanging. Or even if they are

changing, it's not by much. And so maybe you want to remove them from consideration.

That's when you use a technique like dimensionality reduction. And this is really, really important when you're doing feature selection and feature extraction in your real data sets. And you might also use it for other techniques, such as compression or visualization.

So if you want to show things on Excel, showing a thousand dimensional object may be difficult.

You might try to project it down to the two or three dimensions that are easiest to visualize.

And of course, you can use neural networks for unsupervised learning as well. Surprise, surprise.

So as much as a lot of the press you've seen has been on things like image classification using nice labeled data sets, there's a lot of work where you can apply it in an unsupervised case. And these are largely used to find better representations for data, such as clustering and dimensionality reduction. And they're really powerful because of their non-linear capabilities.

So one example-- I won't spend way too much time on this-- is an autoencoder. And the basic idea behind an autoencoder is you're trying to find some compressed representation for data. And the way we do this is by changing the metric that we use to say that the system has done a good job.

And the metric is basically-- if I have a set of input features that I'm passing in, I would like to do the best job in reconstructing that input at my output. And what I do is I squeeze it through a smaller number of layers, which forms this compressed representation for my data set.

And so the idea here is, how can I pass my inputs through this narrow waste to come up with a reconstructed input that's very similar to my original input? And so my metric in this particular case is essentially the difference between the reconstructed input or the output and the input. And the compressed representation-- you can think of as the reduced dimensionality version of my problem.

We've also done some work on replicator networks, which are also really, really cool. Happy to chat about that as well. And finally, we have to talk very briefly on reinforcement learning. And the basic-- again, at a very high level, the reason reinforcement learning is fundamentally different than supervised or unsupervised learning is that you're not passing in a label associated with an input feature.

So there is no supervisor or a person that can label it, but just a reward signal. And the feedback is often delayed. And time is important, so it steps through a process.

And the agent's actions often change the input data that it receives. So just to maybe-- in the interest of time, just to give you examples of where reinforcement learning could work and why you would use a technique like reinforcement learning.

So flying stunt maneuvers in a helicopter. So if your helicopter is straight, you say keep doing more of whatever you're doing to keep it there. If the helicopter tips over, you say stop doing whatever you just did to do that. Could you create a supervised learning algorithm for doing this? Sure. Right?

You would basically look for all the configurations of your entire system every time the helicopter was upright. And you would look for all the examples where your helicopter was tipping over or falling. And you would basically say, OK, my engine speed was this much. My rotor speed was this much.

And there are probably people here who fly helicopters, so pardon me if I am completely oversimplifying this problem here. However, you could certainly label it that way and say all these configurations of the helicopter meant the helicopter was upright. All these configurations of the helicopter meant the helicopter was not upright.

That would be pretty expensive and difficult data-- collect to do. Not sure how many people want to volunteer for-- let's do all the ones that are at faults. And lots of other applications beyond that. So these are really useful, especially in cases where-- what you're trying to model is just extremely complex. And the other really powerful thing is this tends to mimic human behavior. And so they're very useful in those type of applications.

AUDIENCE: Can you explain, shortly, what a reward would look like?

VIJAY GADEPALLY: So a reward would just be-- it would be very similar until you get points.

So you have your algorithm that's basically trying to maximize the number of points that it receives, for example. And as you do-- it's very similar to what you or I would consider a reward playing a video game, right?

Every time I get points, I do more of the activities that make me get points. And it's essentially the same concept over here. All right. So with that, I will conclude only 20 minutes behind schedule. So I guess the long story short is there's lots of exciting research into AI and machine learning techniques out here.

We did a one-hour view of this broad field that research has dedicated about six to seven decades of work to words, so my apologies to anyone watching this or in the room whose work I just jumped over.

The key ingredients, however-- and I think this is most important to this group-- is I look at what are the problems where AI has done really well.

These are some of the key ingredients-- data availability, computing infrastructure, and the domain expertise and algorithms. And I think it's very exciting to see this group over here, because we do have all of these pieces coming together. So great things are bound to happen.

There are, I think, large challenges in data availability and readiness for AI, which is what we're just going to scrape the edge off during this class. And some of the computing infrastructure is something that we'll be talking to you about in a couple of minutes.

And if you're interested in some of the more detailed look at any of these things, a number of us actually wrote-- maybe I'm biased. I think it's a great, great, great write-up.

But, no, I think it's useful. It has its places. Obviously, a lot of material in here. But we try to do our best job to at least cite some of this really, really interesting work that's going on in the field. So with that, I'll pause for any additional questions, but thank you very much for your attention.