[MUSIC]

**PERCY LIANG:** At the end of the day, you know, we're computer scientists building, you know, systems for the world, and I think humans make mistakes. They have fallacies, they have biases, they're not super transparent sometimes, and why inherit all these when, maybe, you can design, you know, a better system? And I think computers, already, clearly have many other advantages that humans don't have.

[MUSIC]

**KEVIN SCOTT:** Hi, everyone. Welcome to Behind the Tech. I'm your host, Kevin Scott, Chief Technology Officer for Microsoft.

In this podcast, we're going to get behind the tech. We'll talk with some of the people who have made our modern tech world possible and understand what motivated them to create what they did. So, join me to maybe learn a little bit about the history of computing and get a few behind-the-scenes insights into what's happening today. Stick around.

[MUSIC]

**CHRISTINA WARREN:** Hello, and welcome to Behind the Tech. I'm Christina Warren, Senior Cloud Advocate at Microsoft.

**KEVIN SCOTT:** And I'm Kevin Scott. Today our guest is Percy Liang. Percy is an Associate Professor of Computer Science at Stanford University, and one of the great minds in AI, specifically in machine learning and natural language processing.

**CHRISTINA WARREN:** Yeah, and Percy talks about the need for AI to be “safely deployed,” and he says that “given society’s increasing reliance on machine learning, it's critical to build tools that make machine learning more reliable in the wild.”

**KEVIN SCOTT:** Yeah, I completely agree with Percy's point of view, and honestly with – like a bunch of his other very interesting ideas about how machine learning and natural language processing are unfolding over the next few years. So, I'm super interested in having this conversation. So, let's find out what Percy's up to.

[MUSIC]

**KEVIN SCOTT:** Our guest today is Percy Liang. Percy is an Associate Professor or Computer Science at Stanford University. He's also one of the top technologists at Semantic Machines. His two research goals are to make machine learning more robust, fair and interpretable, and to make it easier to communicate with computers through natural language.

He's a graduate of MIT and received his PhD from UC Berkeley. Hey, Percy, welcome to the show.

**PERCY LIANG:** Thanks for having me.

**KEVIN SCOTT:** So, we always start these shows with me asking how you first got interested in technology. Were you a little kid when you realized that you were interested in this stuff?

**PERCY LIANG:** Yeah. I think it was around maybe end of elementary school or middle school. My dad always had a computer, so it was around, but he didn’t let me play with it --

**KEVIN SCOTT:** And what did your dad do?

**PERCY LIANG:** He was a mechanical engineer. Yeah. And I remember maybe my first memories are in that first school -- in middle school, there was a computer lab and there was a HyperCard, which is this multimedia program for the Macintosh back then. And I got really fascinated in building these relatively simple applications, but they had a scripting language so you could start to code a little bit. And there is animation and all that. So it was kind of fun to get into that.

**KEVIN SCOTT:** I remember HyperCard as well. I believe one of the first programs I wrote -- I may be a little bit older than you are, but I do remember at one point writing a HyperCard program that was a multimedia thing that animated a laser disc. Do you remember laser disks, the big, gigantic precursors to DVDs? Yeah, it was really such a great tool.

**PERCY LIANG:** Yeah. At that time, I also tried to learn C, but that was kind of a disaster -- (inaudible) pointers and all this stuff.

**KEVIN SCOTT:** Yeah, C is sort of a formidable first language to attempt to learn. One of the things, given that you are a computer science educator, I’d be curious to hear how you think about that evolution of entry into computer science. On some levels now, it seems like it’s a lot easier to get started than when we were kids, maybe. But in other ways, it’s actually more challenging because so much of the computing environment -- like the low-level details -- are just abstracted away and the layering is very high and it is a lot to get through.

**PERCY LIANG:** Yeah. Somehow, computer science thrives on abstraction, right? From the low-level machine code to C and we have Python and programming languages and at some level, graphical interfaces. So picking the right entry point into that for someone is -- I think there are multiple ways you can go.

I probably wouldn’t start with C if I were teaching an intro programming class, but more at a conceptual level of here are the kinds of computations that you want to perform. And then separately, I think a different class would talk to you about how this is actually realized because I think there is some value for a computer scientist to understand how it goes all the way down to machine code, but not all at once.

**KEVIN SCOTT:** Yeah, I’m still convinced that one of the most useful things that I had to learn as a programmer who learned to program in the ‘80s was fairly quickly I had to learn assembly language. You had to know what the low-level details were of the machine.

Now, granted, the machines were vastly less complicated back then than they are now. But just sort of at that atomic level, knowing how the actual machine works just made everything else that came after it less intimidating.

**PERCY LIANG:** Yeah, it’s kind of satisfying. You’re grounded. It’s like playing with blocks almost.

**KEVIN SCOTT:** So, you started with HyperCard. Where did things go from there?

**PERCY LIANG:** Yeah, for a while, I think I also learned BASIC, I was just kind of tinkering around. There wasn’t like today, as many resources, as you can imagine, for just kids interested in programming. So a lot of it was on my own.

I think maybe a turning point happened at the beginning of high school, where I started participating in this USA Computing Olympiad, which is a programming contest. You can think about it as a programming contest, but I really think about it as a kind of algorithmic problem-solving contest.

The problems that they give you are-- it’s kind of like a puzzle and you have to write a program to solve it. But much of the work is actually kind of coming up with the insight of what algorithm to do it kind of efficiently. An example might be how many ways are there to make change for $2 using a certain set of coins?

And it would be this eureka moment when you found, “Ah-hah, that’s how I can do it.” And then you have to code it up. So I think that competition really got me to value this type of rigor and attention to detail, but also the creative aspect of computing, because you have to come up with new types of solutions.

**KEVIN SCOTT:** That’s awesome. What was the most interesting problem you had to solve in one of these competitions?

**PERCY LIANG:** That’s a really good question. It’s been a while, so I don’t remember all the problems. But one I think -- one memorable maybe class of problems is around the idea of dynamic programming -- this idea that you can write a program -- if you do it smartly, you can make something that would otherwise run in years or millennia in a matter of seconds.

I remember having to -- it was always these problems, and you had to really figure out what was the recurrence relation to make it all work? And a lot of problems were kind of centered around --

**KEVIN SCOTT:** Yeah, one of the amazing things about the dynamic programming technique is it really does teach you -- and it might be one of those foundational things when you’re getting your head wrapped around how to think algorithmically about problem decomposition. Because it’s one of those magical things where if you break the problem down in just the right way, all of a sudden a solution to the problem becomes possible when it was intractable before.

**PERCY LIANG:** Yeah. I think I liked it because it wasn’t that you had to memorize a bunch of things or you learned -- if you learn these 10 algorithms, then you would be set. But it was kind of much more open-ended way to think about problem solving.

**KEVIN SCOTT:** Yes, that’s awesome. And so you go to MIT as an undergraduate student? How soon did you know exactly the thing inside of computer science that you wanted to do?

**PERCY LIANG:** That, I think, took a little bit of evolution. Coming out of high school, I was much more interested in these algorithmic questions and got interested in computer science theory because that was kind of a natural segue. And I started doing research in this area. It wasn’t until toward the end of my undergrad where I started transitioning into machine learning or AI.

**KEVIN SCOTT:** And when was this? What year?

**PERCY LIANG:** This was around 2004, yeah.

**KEVIN SCOTT:** Still, machine learning was --

**PERCY LIANG:** Yeah, people didn’t use the word “AI” back then.

**KEVIN SCOTT:** Yeah, I mean, I remember like right around that time was when I joined Google and I had been a compiler guy when I was an academic. I had never done AI at all. I didn’t know what machine learning was when I started and, yet, three months after I joined Google, I was tasked with doing a machine learning thing and reading this giant stack of papers and formidable textbooks trying to get myself grounded.

**PERCY LIANG:** Yeah.

**KEVIN SCOTT:** It’s a very interesting time, 2004, and you picked a great time to get interested in machine learning.

**PERCY LIANG:** Yeah, I had no idea -- no idea that it would be the field that it is today.

**KEVIN SCOTT:** And why was that interesting? So, like I can sort of get why the theory was interesting, like you loved these problems and the challenge of them. What was interesting about machine learning?

**PERCY LIANG:** I mean, I think there’s definitely this background -- AI would be kind of mystical aspect of intelligence that I think I’m not unique in being drawn to. When there was an opportunity to connect the things that I was actually doing with the theory with some element of that, I took the opportunity to kind of get into that.

And then I stayed at MIT for my Master’s, which was on machine learning and natural language processing. Then, that kind of really cemented the direction that I really started pursuing.

**KEVIN SCOTT:** What -- like I’m sort of interested because if you did your master’s degree there, this was right before the deep learning boom. So, it wasn’t the same flavor of machine learning, natural language processing that folks are very excited about right now, so like --

**PERCY LIANG:** That came quite a bit later.

**KEVIN SCOTT:** What was your thesis about in particular?

**PERCY LIANG:** Yeah, my thesis was actually at MIT was about semi-supervised natural language processing. So, in some ways, there are spiritual connections to a lot of the things like BERT and these things that you see today -- the idea that you can use a lot of unlabeled data, learn some sort of representations -- those were based on this idea called “brown clustering.” And that was used to, then, improve the performance on a number of tasks. Of course, data sets and compute and all the regimes were different, but somehow the central ideas have been around for a while.

**KEVIN SCOTT:** Yes. And so what did you do your dissertation on?

**PERCY LIANG:** Yeah, so during my PhD at Berkeley, I did a bunch of different things ranging from more theoretical machine learning to applied natural language processing. But toward the end of the PhD, I really kind of converged on semantics or semantic parsing as a problem. How do you map a natural language utterances into some sort of executable program or meaning representation?

An example is if you have a database of U.S. geography, you can ask, “What’s the tallest mountain in Colorado?” It would translate into a little program that perform the database query and deliver you the answer. Yeah.

**KEVIN SCOTT:** Right. And the challenge there is like you might have a database that’s got like a whole bunch of geographical objects in them and like you have a type which might be “mountain” and like the thing might have a height property and, like, and it’s all described in this very exact way. And, like, the human utterances are very inexact sometimes.

**PERCY LIANG:** Exactly. Yeah. So the main challenge behind all of natural language processing, no matter what task you take, is just the fluidity of human language. You can say something -- the same thing in many different ways and there are nuances. I could ask, “What’s the tallest mountain in Colorado? In Colorado, what’s the highest mountain?” And so on. Having to deal with that ambiguity is, I think, the value proposition of natural language processing.

**KEVIN SCOTT:** Interesting. When you were finishing up your degree, did you know that you wanted to be a professor at that point?

**PERCY LIANG:** Yeah. Because the exact research area I think was still a little bit up in the air. I was having a lot of fun with the semantic parsing problem. Then I spent a year in Google actually working on a semantic parser for them that powers a lot of back then was Google Now.

**KEVIN SCOTT:** So, they have a semantic parser with the funniest name in existence, this thing called “Parsey McParseace.” That wasn’t your thing, was it?

**PERCY LIANG:** That was later, yeah.

**KEVIN SCOTT:** Yeah. (Laughter.)

**PERCY LIANG:** That was --

**KEVIN SCOTT:** Which is a very silly name for a parser.

**PERCY LIANG:** It’s very memorable, yeah. (Laughter.)

**KEVIN SCOTT:** But you can well imagine how this technology might be super important in search where the whole search problem is asking questions of a search engine and the search engine needs to understand something about the question so that it can get reasonable answers.

**PERCY LIANG:** Yeah. Search, assistance, and all these cases where there is a human with some information need or some action that needs to be taken, the most natural way is to use, well, natural language and how to get computers to understand that to the extent of being useful delivering something useful to the user is the central question.

**KEVIN SCOTT:** So, you got your PhD at Berkeley and then what happened next?

**PERCY LIANG:** I applied to jobs. I got a position at Stanford, which I was very happy about. Then I took a year off -- I mean, in quotes, a year off to do something different. And I knew I was going to be a professor and write papers, so I wanted to have -- see how I could take this technology and actually make it kind of real in some sense.

So, I did a post-doc at Google. And was trying to figure out how to use semantic parsing for something. And at that time, so this was 2011, I think Siri had just come out for the first time. I think there was a sense that inside Google that -- well, we should do something big about this.

And so other people and I formed a team and we built the semantic parser that then powered relatively simple commands, but then increasingly, over time, got to powering questions and all sorts of other things. So that was really exciting to see how the tech transfer happens from academic research to actual products.

**KEVIN SCOTT:** Explain to folks like how it’s different. Like, building a product where it just sort of has to work all the time for all of the users is sometimes different from building a thing that’s good enough to write a paper about.

**PERCY LIANG:** Yeah, definitely. I think there is quite a big gap between what counts as a product and what counts as a paper. The desiderata are also different, right? I think in academia, the currency is intellectual ideas. Do you have something interesting to say?

A lot of the techniques actually are interesting, but they aren’t really ready to be deployed because they don’t work nearly well enough. And if you’re launching a product, it has to work, like you said, 99% of the time at least. And it can’t make embarrassing errors and it has to be fast and usable.

So I think there’s a lot of pieces that have to go into making a product. Also, in academia, people work on data sets, but the data sets are insufficient to represent the diversity of things that you would see in the real world. So that’s something that needs to be solved as well.

I think there’s actually a lot of interesting research problems around the kind of ecosystem of product deployment, which are not so much the focus of academic research probably because it’s actually hard to get an idea of that ecosystem, but it’s super valuable.

**KEVIN SCOTT:** So did you ever -- like either yourself or the teams that you work with struggle with this split between like the sort of very intellectually interesting and challenging part of building a product versus the very like, you know, sort of mundane, “grunty” part of building a product?

**PERCY LIANG:** So, at that time, I wasn’t interested in writing a paper, I just wanted to kind of execute. I don’t think there was so much of that tension. It was just “do whatever it takes to get this out the door.”

**KEVIN SCOTT:** It’s super interesting because I’ve had -- I’ve managed teams of people doing machine learning work and -- who had PhDs in machine learning. And like the thing that attracted them to machine learning in the first place is they were interested in like the core research, like the challenging problem, like, how to make this very complicated thing, like, you know, one epsilon better than what preceded it, and who got frustrated very quickly with what production machine learning looks like, which is more like lab science than it is like theoretical computer science, for instance.

Sometimes I’ve had -- I’ve had people who, you know, like, on paper look super qualified because they have written a dissertation on machine learning to work on an ML team where, you know, someone who, like, has a degree in applied physics, for instance, is much more excited working on the machine learning problem because, like, they are more interested in this sort of iterative, like, approach to -- you’re wrangling the data and doing experiments and whatnot.

**PERCY LIANG:** Yeah.

**KEVIN SCOTT:** So, it’s great that you -- like, you never felt that tension, like, that’s almost a superpower. (Laughter.)

**PERCY LIANG:** Yeah, I mean, I think at some level, I’m interested in solving problems. And I think there’s actually, in my head, there’s sometimes even a deliberate dichotomy between-- What am I trying to do? Am I trying to build a system that works or am I trying to understand a fundamental question?

And sometimes research can get a little bit muddled, where it’s not clear what you’re trying to do. I have some more theoretical work which has no direct implications on product, but it’s just so intellectually stimulating that you pose this question and you try to answer it.

**KEVIN SCOTT:** And do you think that’s one of the, like, benefits of academic research, like, doing what you do in a university versus a company where you’ve got the freedom to have this mix of these multiple things that you’re pushing on?

**PERCY LIANG:** Yeah, definitely, I feel like the benefits of academia are -- is the freedom. I feel, you know, pretty much full freedom to think about what are the ideas that I think are interesting and pursue them.

I think also, students come into the picture quite heavily because they’re the ones also contributing and thinking about the ideas collectively with me. Yeah, I think it’s really, you know, an exciting environment.

**KEVIN SCOTT:** So, like, back to your story. So, like, when did you decide to do semantic machines?

**PERCY LIANG:** Yeah, so I started at Stanford in 2012. And for the first three years or so, I was just trying to learn how to be a professor, teach classes, advise students. So there’s plenty of stuff to do. I wasn’t looking to join a startup.

But then around 2016, so Dan Klein, who was one of my advisors at Berkeley, came to me and he was working on Semantic Machines, which I had known about. And basically convinced me to join.

And I think it was a, you know, and I think the reason for doing so is, you know, if I think about my experiences at Google, where you would take ideas and you kind of really get them to work in practice, I think that it was kind of a very, you know, compelling environment and Semantic Machines had a lot of great people, some of which I knew from grad school. And I think the kind of critical mass of talent was, I think, one of the main draws because you have a lot of smart people working on this incredibly hard problem of conversational AI or dialogue systems.

Yeah, so it was kind of irresistible. Even though, you know, my sanity probably suffered a little bit from that. (Laughter.)

**KEVIN SCOTT:**  So, what are -- what are some of the big challenges that we still have open in conversational AI, so like you're trying to build an agent that you can communicate with as, as you would another human being. So, like some things are -- they're like really great, like speed track ignition, like turning the utterances that come out of your mouth into some sort of structured representation, like that's pretty good now. But like there's still some big open problems, right?

**PERCY LIANG:** Yes, there are a ton of open problems. I'm not worried about losing my job any time soon. I think the -- maybe the way to think about it is that the history of NLP has always been kind of this tension between breadth and depth, right? We have, in the '80s and '70s, very deep language understanding systems, and the domains, and you could ask all sorts of questions, and we'll do a good job.

But, you know, once you go out and leave the confines of that domain, then all bets are off. And on the other hand, we have things like search which are unstructured, they're just broad. They don't claim to understand, in any sense of the word, understand anything, but they're incredibly useful, just because they have that, you know, breadth.

And I think while there's still a huge gap between -- and the open challenge on how do you kind of really marry the two, and a lot of these kind of conversational systems where you actually have to do things in the world, not just kind of answer questions, are -- do require some amount of structure, and how do you marry that with the kind of open-endedness of something like search, is --

**KEVIN SCOTT:** And just to like, just the way that I think about those two ends of the spectrum, right, is you have these like structured dialogue systems where you have to ask the question in exactly the way, or like pretty close to exactly the way the system expects you to ask the question, in order for it to be able to respond.

And on search, you can get a broad range -- you can ask the question, like a bunch of different ways, and like expect to get a response because the question has been answered in like a gazillion possible ways, on the web, and like you're going to get, you know, maybe one of those answers returned to you.

And like the hard part is like in-between, is of like something really understanding the question that you're asking, or the command that you're giving to the system, and like understanding it enough so it can then go like connect to whatever knowledge repository or set of APIs, or whatever else, that is going to do the thing that they want done.

**PERCY LIANG:** I mean, one -- the one thing that search, I think, did really well is the interface, right? The interface promises nothing. You, you -- it promises template links or maybe some summaries, and I think, as opposed to in a system, where it's just framed as an AI who is trying to do the right thing for you, and there's only disappointment when it doesn't, whereas as search, how many times you search and you don't find what you want.

And it's like, okay, well, it's user error, let's try again, but that's -- allows you to get so much more data and signal, and a potential for improvement, which -- whereas if you have an assistant and it just doesn't work, then you just, you're just going to give up.

**KEVIN SCOTT:** Yes, there is this weird psychology thing, right, where -- with the interfaces, like you almost feel embarrassed when you ask the -- like the software question, verbally, and it doesn't give you the right answer, like you just sort of assume that you've done something wrong, whereas somehow or another, with search, like we've -- and like it's, it reminds me a little bit of my mother.

Like my mother, whenever she can't get her computer to do what she wants it to do, she always assumes that it's her fault, which is a weird way to approach technology.

So, let's go back to the work that you do at Stanford. So, you spend part of your time teaching students, like in particular, like you're teaching the -- some of the AI curriculum at Stanford, and then you're doing, you're doing research. So, talk a little bit about the teaching, like how has teaching students machine learning changed over the past handful of years?

**PERCY LIANG:** Yeah, so I -- the main class that I teach at Stanford is CS221, the main AI class, and I've been teaching this since 2012. When it started there were less than 200 students in the class, and last year there were 700 or so. So, there's definitely-- the most salient thing that has happened is just the sheer number of students wanting to learn this subject matter.

So, that has presented a number of challenges. I think people are taking the class from a fairly heterogeneous population. There's undergrads who are learning computer science and trying to -- you know, are excited about AI, and they want to learn about it. There's Master’s students who have a little bit more research experience, maybe.

There's people from other departments, who have actually quite advanced mathematical abilities, and are trying to learn about, you know, AI. There are people, professionals who are working full-time and trying to learn about AI. So, one of the challenges has just been to how to accommodate all these -- this diverse population.

**KEVIN SCOTT:** And how do you do that?

**PERCY LIANG:** It's, it's challenging, there are certain things that, you know, that we try to do, trying to have materials which are presented from kind of slightly different perspectives, and have, you know, review sessions on certain types of topics. But, honestly, I don't have a, you know, great solution. We have a lot of TAs who can, you know, help, but it's, I think scaling education is one of these very hard problems.

**KEVIN SCOTT:** Yes, like when I was teaching computer science, when I was working on my PhD, the thing that was always super challenging, for me, like I was -- I taught CS201, I think a couple of times, which was -- like at the University of Virginia, it was the first serious software engineering course that you took, or programming course.

And like we had such a broad range of students taking the class that it was, and you would have people who came in who were -- like had years and years of experience, like by the time they got their programming. Like they learned to code when they were 12, and you sort of risk every other thing you were doing, boring these poor kids to death.

And then you had folks who were like coming in because they were interested in computer science, and like they had almost no background, whatsoever, they never programmed, and like they might not even have the -- you know, sort of the analytical, you know, background that is helpful when you're learning to code. And like that was always a huge challenge for me, and like I don't know whether I was ever any good at it, or not--

**PERCY LIANG:** Yeah, I think that if I had much more time, I would kind of sit down and really think about how to best structure this. I mean, I think the -- I think the way to do it is trying to break things down into modules and making sure that people understand basic things before they move on to more advanced things. I think when you have these kind of banner courses like AI, people take it, but they don't really --they land somewhere in the middle, and they're trying to figure out things, and it's a much more of a kind of a treading water kind of a situation as opposed to like really kind of building up, you know, building blocks.

**KEVIN SCOTT:** So, one of the interesting things that I think has really happened over my career doing machine learning things, is -- in 2003, when you're doing machine learning stuff, like you're more or less starting from scratch, whenever you're trying to build a system, and like now, if you want to do something with machine learning, you've got PyTorch, you've got -- you know, you've got like notebooks, like Jupyter Notebooks, you've got like all of this sort of incredible infrastructure that is available to you, to like build things.

Like my favorite anecdote is like the thing that I did at Google, which is my first machine learning project, that took like a, reading a bunch of like heavily technical stuff, and like probably six months' worth of very hard work, like a high school kid with sufficient motivation, like using a bunch of opensource tools could do in a weekend, which is just incredible.

But I'm guessing that also puts pressure on the curriculum, like what you provide as programming exercises for kids, where you just sort of -- just keeping pace with the overall field, it's got to be challenging, right?

**PERCY LIANG:** Yes, so it's certainly very incredible how far we've come, in terms of tools. And, again, this is kind of the success story of abstractions in computer science where we don't -- many people don't have to think about registers to program, and get even close to kind of assembly, although people program Python might not have to think about, you know, memory management, and now, when you're working with something like PyTorch or TensorFlow, you can think about the modeling, and focus on the modeling without thinking about how the training works.

You know, of course, I think in order to get off the ground and have kind of a hackathon project, you can get by with not knowing, you know, very much. I think to get kind of really kind of serious, the -- these abstraction barriers are also leaky, and I think someone would be well served to understand, you know, how -- what are gradients and how that is, you know, computed.

So, I think in the class that I teach we definitely expose students to the raw, kind of the bare metal, so to speak. For example, in the first class I show people how to do stochastic gradient descent, and it's -- I, you know, code it up, and it's 10 lines of code, and not using PyTorch and TensorFlow.

And I want people to understand that some of these ideas are actually pretty, pretty simple, and -- but, you have to kind of -- but, I wanted people to get exposed to the simplicity, rather than being kind of scared off by, oh, that's underneath the PyTorch wrapper, and because at some level all of these pieces are actually quite, you know, understandable.

**KEVIN SCOTT:** Yeah, and I think that's a great thing that you're doing for your students because one of the things that I do worry about, a little bit, is that we have these very powerful abstractions, but the abstractions make a bunch of assumptions that are not necessarily correct, that for instance, that stochastic gradient descent is the best numerical algorithm to fit the parameters of a deep neural network.

It's very good, the technique, but like we shouldn't assume that that is a solved problem. Like, in fact, there was this paper NeurIPS, a couple of years ago, on -- I think the title was "Neural Ordinary Differential Equations," where they were like modeling the interior state of a DNN and using ordinary differential equations, and using -- I think something like fourth order Runge-Kutta, or something to solve it, which is very, very, very different from -- you know, stochastic gradient descent. And like the fact that -- like that sort of exploration is great that it's still happening.

**PERCY LIANG:** Yeah, one thing that I do in the AI class is be very kind of structured about the framing of a class, in terms of modeling and algorithms, right? So, you can think about, for a given problem, how do you construct the model? It could be a neuronal architecture, but it could -- I talk about some other topics like graphical models. It could be like what your vision network looks like.

And then, separately, you think about how I'm going to, you know, perform inference or do learning in these type of models. And I think that the coupling is something that I find students often kind of find it hard to think about because, knee -- your knee-jerk reaction to solve a problem is just go directly solve a problem, but figuring out how to model the situation, which specifies kind of what you want to do, and then the algorithms are how you want to do it, is really I think a powerful way to, you know, think about the world.

**KEVIN SCOTT:** So, as an NLP person what do you think about all of this stuff happening with self-supervised learning, right now, and natural language processing? So, this is the BERTs, the GBT-2s, the ExcelNet!s, like we've even, you know, just a little while back, like Microsoft disclosed this new, like Turing-NLG, which is a 17-billion parameter model that we're being -- that we've been working on, for a little bit.

**PERCY LIANG:** Yeah. No, it's super impressive. I mean, I would have said that, you know, four or five years ago I wouldn't have predicted kind of the extent to which these things have been successful, and it's certainly not the idea of doing so. I mean, these are things that I've even explored in my Master's thesis, but that's clearly a big difference between having an idea and actually showing that it actually, you know, works.

So, clearly, these methods are being deployed everywhere, and I think people are getting quite a bit of mileage out of them. I think there's still problems that these methods are not -- you know, sufficient to solve by themselves. I mean, I think they're going to be part of probably any NLP solution for -- until the end of time, but I think kind of deeper language understanding, beyond kind of these -- the benchmarks that we have, are going to possible demand some other ideas.

**KEVIN SCOTT:** Yeah, and that certainly seems true, even though I'm very bullish about like the fact that we seem to be able to get performance, like improving performance by making the models bigger on the things that the models are good at. It's still unclear to me that they're going to be good enough at everything. Like this, this is not going to solve AGI in and of itself, I don't think.

**PERCY LIANG:** Yeah, I mean, it's interesting to ponder how far you can push these, like if you train it on literally all the text in the world, you know, what do you get?

**KEVIN SCOTT:** And you gave it as many neurons as the human brain?

**PERCY LIANG:** Yeah, yeah.

**KEVIN SCOTT:** We're likely to find out at some point.

**PERCY LIANG:** I think there is cases where -- you know, we've been doing it in our research group at Stanford where, even the most kind of advanced models make the -- kind of the most -- the dumbest mistakes, where you have a question answering system, you add an extra comma, or you replace a word with a synonym, and it goes from working to not working.

So, even with BERT, and things like this. So, it's kind of interesting to ponder the, you know, significance of that. So, from a practical perspective, you know, it doesn't matter, actually, that much, because you -- through more data, you can kind of get things to work, and on average, things will be fine, but from kind of an intellectual perspective of do these models really understand language, the answer is kind of a clear -- at least, for me, a clear no because no human would make some of these mistakes.

**KEVIN SCOTT:** Yeah, although, that's an interesting thing that I've been thinking about, and I think that I actually agree with you, but like one of the things that I've been pondering, the past few months, is just because these models -- and it's not just the speech models, like vision models, also, you like stick a little bit of what looks like uncorrelated noise into them, and all of a sudden -- like, you know, you've recognized my face is like my boss' face, right?

Like they make mistakes in ways that are very idiosyncratic to the models, and very, very much not like the mistakes that humans would make. But humans make mistakes, as well, and I just sort of wonder whether -- like for myself, that I am creating an unnecessary false equivalence between these AI systems and like biological intelligent systems, where just because it makes a -- the software makes a mistake that a human wouldn't, doesn't mean that it's not doing useful things.

And like, just because -- you know, like I can solve problems easily that it can't, you know, doesn't invalidate the thing that -- you know, that the machine learning system can do.

**PERCY LIANG:** Yeah, definitely. I think these examples merely illustrate the kind of the gap between these machine learn models and humans, and I think it's absolutely right to think of machine learning AI as not chasing human intelligence, but more -- they're a different thing. And I've always thought about these things as -- you know, tools that we build to help us.

I think they're -- a lot of AI does come from this, you know, chasing human intelligence as inspiration, which has gotten, you know, quite a bit of mileage. But at the end of the day, you know, we're computer scientists building, you know, systems for the world, and I think humans make mistakes. They have fallacies, they have biases, they're not super transparent sometimes, and why inherit all these when, maybe, you can design, you know, a better system?

And I think computers, already, clearly have many other advantages that humans don't have. They have, you know, they don't need to sleep, they have -- you know, memory which is vast and compute that vastly exceeds --

**KEVIN SCOTT:** They don't get bored.

**PERCY LIANG:** Yeah, yeah, so I think leveraging these, which we already have, but kind of further just thinking holistically about how do we build the most useful tools might be a good way forward.

**KEVIN SCOTT:** Yeah, yeah, I really love that vision, like this sort of notion of thinking about the AI systems as tools, like maybe thinking more about task intelligence than general intelligence, and you know, trying to derive inspiration from biology, but like not being, you know, not being fixated on it.

**PERCY LIANG:** Yeah, I mean, this kind of whole debate goes back to, you know, the '50s, and with AI versus IA, artificial intelligence versus intelligence augmentation, where intelligence augmentation kind of more -- we're, as kind of a spiritual ancestors of, you know, the field of HCI, human computer interaction, and at some level I think that I'm more kind of philosophically attached to that kind of way of thinking about how we build tools. But clearly AI is providing this kind of massive set of, you know, assets that we should be able to use, somehow.

**KEVIN SCOTT:** Yeah, so a couple more questions before I ask you something, something fun. So, how do you think academia and industry could be doing more together? Like one of the things that I'm a little bit worried about, like less so this year, than last, is some of these machine learning workloads now just require exorbitant levels of resources to run. So, like training one of these big self-supervised models, just like -- the dollar cost on the computations is getting to be just gigantic, and it's going to grow.

Like it's been expanding at about, you know, 8-10x a year, and so I sort of worry about -- like with this cost escalating, like how can everybody participate in the development of these systems, like especially universities, like even well-resourced ones like Stanford?

**PERCY LIANG:** Yeah, I think it's like on a lot of people's mind, the compute required for being kind of relevant in the kind of modern ML. I think there's a -- there's a couple of things. What's certainly -- I know that companies have been providing, you know, cloud credits to academia, and certainly this has been helpful, probably more of it would be more helpful, and -- but, I think that's maybe not a -- you know, in some sense kind of a panacea because what -- however many cloud credits industry gives, industry is always going to have -- you know, more of that they can do in-house.

But I think a lot of the way that I've been thinking about research at Stanford, without unlimited resources is, a lot of times, you can be kind of orthogonal to what's going on. So, so some of our recent work focuses on methods for understanding what's going on in these BERT pretrained models, or how to think about interpretability or fairness.

And I think some of these questions are fairly kind of conceptual, and the bottleneck there isn't just doing more compute, but to actually even define the question, and think about how to -- you know, frame it and solve it. And I think that another thing which I alluded to earlier is that there are clearly a lot of real-world problems that, you know, industry is facing, not just in terms of scale, but the fact that there's real systems with real users.

There's feedback loops, there's biases, and heterogenic -- heterogeneity, and I think there's a lot of potential for surfacing these kinds of questions that I think that the academic community would be -- would be helpful in kind of answering that, not at a -- kind of a conceptual level. I think probably teams are probably too busy to be worried -- pondering about what is the right way to solve these problems, but they have the problems, and it these can be somehow, you know, brought out, I think we would probably be able to leverage all the kind of intellectual horsepower in academia to solve kind of real -- really relevant, you know, problems.

**KEVIN SCOTT:** That sounds like a great idea. So, two more questions. One, one, in your role as an AI researcher, so what's the -- what's the thing that excites you most about what you see on the horizon, like what's going to be really interesting over the next few years?

**PERCY LIANG:** Yeah, if only I could predict the future. I think one of the things that has been exciting to me is program synthesis, the idea that you can automatically write programs, from either test case examples, or natural language, and in some ways, this is kind of an extension of some of the work I did on, you know, somatic parsing, but if you think about it, from -- at a high level, you know, we have users, and they have, you know, desires and things that they want to do, how can you best harness the ability of a computer to, you know, meet those needs?

And currently -- while you have -- you can either program, if you know how to program, or you can use one of these existing interfaces, but I think those kind of two are, you know, very limiting. If, if you could have users that could kind of express their desire in some sort of more fluid way, even with examples or language, and computers kind of synthesize these programs or computations, then you could really, I think, amp up the amount of leverage that ordinary people have.

And also to think about how even not kind of end users, but, you know, programmers could benefit from a lot from having, you know, better tools.

We have these enormous code bases and programming is, at the end of the day, a lot of in-the-weeds work. And I think, you know, the use of machine learning and program synthesis could really open up the way towards maybe a different -- a completely different way of thinking about programming and code, and that's kind of -- as a computer scientist, that is very, you know, fascinating.

**KEVIN SCOTT**: Yeah, and I'm, I'm really glad to hear you say that you're excited about the prospect of that because one of the things that I do worry about is we -- you know, we're now at the point where non-tech companies are hiring more software engineers than tech companies, like it really is the case that like every company, like has to deal with code and software, and like the value that they're going to create over the next several decades of their business is going to be in this sort of -- like IP, and like software artifacts that they're creating to run their businesses and solve their customers' problems.

And like they're just aren't enough programmers on the planet to go do all of this work, and like a lot of our customers, like especially when you're talking about machine learning, like they, they just can't hire -- like we, we have a hard enough time hiring all the people in the tech industry in Silicon Valley, right, and so this idea that, like we could change the paradigm of computing to be -- like we all know how to teach our fellow human beings how to do things, like if you could figure out how to teach computers how to do things, on your behalf, like that then opens things up to like an unbelievable number of people to do an unbelievable number of things.

**PERCY LIANG:** Right, yeah, I want to blur the line between what a user and a programmer, and also, it's a really hard problem. Like the best technologies that we have can maybe synthesize 20 lines of code, but if you think about the types of code bases that we're dealing with, it's millions, tens of millions of lines.

So, I think, you now, as a researcher, I'm kind of drawn to these challenges that, where you might need a -- kind of a different insight to make progress.

**KEVIN SCOTT:** Super cool. So one last question. So, just curious what you like to do outside of work. I understand that you are a classical pianist, which is very cool.

**PERCY LIANG:** Yeah, so, piano has been something that's always been with me, since I was -- as a young boy, and I think it's also been a -- kind of a counterbalance to all the other kind of tech-heavy activities that I have been doing.

**KEVIN SCOTT:** What's your favorite bit of repertoire?

**PERCY LIANG:** I like many things, but late Beethoven is something I really enjoy. I think this is where he becomes kind of very reflective about -- and his music has a kind of an inner -- it's, it's very kind of deep, and so I kind of enjoy that.

**KEVIN SCOTT:** Like what, what particular piece is your favorite?

**PERCY LIANG:** So, so there has a -- Beethoven sonata. So, I've played the last three Beethoven sonatas, so Opus 109, 110, 111 --

**KEVIN SCOTT:** Wonderful pieces.

**PERCY LIANG:** Yeah, and one of the things that, actually, you know-- one of the challenges has been it's incredibly hard to make time for kind of a serious habit. And actually in graduate school I was -- I was, very -- there was a period of time when I was really trying to enter this -- or enter this competition and see how well I could do.

**KEVIN SCOTT:** Which competition?

**PERCY LIANG:** It was called the, the International Russian Music and Piano Competition. It was in San Jose, and I don't know why they had this name, but then, you know, I practiced a lot. There's some days I practiced like eight hours a day, but -- and then I was just like this is -- it's just too hard. I can't compete with all these people who are kind of the professionals.

And then it kind of -- I was thinking about how -- what is the bottleneck? Often, I have these musical ideas and I know what I should sound like, but you have to do the hard work of just actually doing the practicing, and you know, kind of thinking maybe wistfully, maybe machine learning AI could actually help me in this endeavor.

Because I think it's kind of an analogous problem to -- idea of, you know, having a desire and having a program being synthesized, or an assistant doing something for you. I have a musical idea, how can computers be a useful tool to augment my inability to find time to practice?

**KEVIN SCOTT:** Yeah, and I think, I think we are going to have a world where computers and like machine learning, in particular, like are going to like help with that human creativity, but like one of the things -- I find classic piano is like this very fascinating thing because, on -- it's one of those disciplines and like there's several of them where it's just blindingly obvious that -- the difference between expertise and non-expertise, like no matter how much I understand --

And so, like I'm not a classical pianist, like I'm just an enormous fan. Even though I understand the -- I understand harmony, I understand music theory, I can read sheet music, I can understand all of these things, and I can appreciate Martha Argerich playing, you know, Liszt Piano Concerto No. 2 at the Proms.

There's no way that I could sit down at the piano and like do what she does because she has put in an obscene amount of work, training her neuromuscular system to be able to play, and then to just have years and years and years of like thinking about how she turns notes on paper into something that communicates a feeling to her audience. And it's like really just to me, stunning, because there's just no -- there's no shortcutting it, like you can't cheat.

**PERCY LIANG:** Yeah, it's kind of interesting because, in computer science, there's oft -- sometimes an equivalence between the ability to generate and the ability to kind of discriminate and classify, right? If you can recognize something, whether it's something's good or bad, you can use that as an objective function to -- to hill climb. But it seems like in music, we're not at the stage where we have that equivalence.

You know, I can recognize when something is good or bad, but I don't have the means of, you know, producing it, and some of that is physical, but I don't know, maybe -- maybe there's a -- this is something that is in the back of my mind, in the back pocket, and I think it's something that -- you know, maybe in a decade or so I'll revisit.

**KEVIN SCOTT:** The other thing too that I really do, I wonder about, with performance, is there's just something about -- like, for me, it just happens to be classical music. I know other people like have these sorts of emotional reactions to rock or jazz or country music, or whatever it is that they listen to. But I can listen to the right performance of like Chopin's G Minor Ballade, and like there are people who can play it, and like -- I 'm like this is, you know, very nice, and like I can appreciate this.

And there's some people who can play it, and it like -- every time I listen to it, 100% of the time I get goosebumps on my spine, like it provokes a very intense emotional reaction. And I just wonder whether part of that is because I know that there's this person on the other end, and they're in some sort of an emotional state, playing it, that resonates with mine, and whether not -- like you'll ever have a computer be able to do that?

**PERCY LIANG:** Yeah, that's a -- I mean, this gets kind of a philosophical question at some point, if you didn't know if it was a human or a computer, then what kind of effect would it have, but --

**KEVIN SCOTT:** Yeah, and I actually, you know, I had a philosophy professor in undergrad who like asked the question, like would, would it make you any less appreciative of a Chopin composition knowing that he was being insincere when he was composing it, like he was -- you know, doing it for some flippant reason, and I was like, yeah, I don't know, like it's --

**PERCY LIANG:** Well, like one of my piano teachers used to say that you kind of have to -- it's kind of like theater. You have to convey your emotions, but there has to be some -- even when you go wild, there has to be some element of control on the back because you need to kind of continue the thread, and --

**KEVIN SCOTT:** Yeah, yeah, for sure.

**PERCY LIANG:** Yeah, but in -- also, it is, for me, also just back to playing, it's just the pleasure of -- it's not just having a recording that's -- that sounds good to me.

**KEVIN SCOTT:** Yeah, no, I'm very jealous that you like had the discipline and did all the hard work to like put this power into your fingers. It's awesome. Well, thank you so much for taking the time to be with us today. This was a fantastic conversation, and I feel like I've learned a lot.

**PERCY LIANG:** Yeah, thanks for having me. My pleasure.

**KEVIN SCOTT:** That's awesome.

[MUSIC]

**CHRISTINA WARREN:** So, that was Kevin's chat with Percy Liang, from Stanford University. And Kevin, you know what was really interesting was hearing both you and Percy reminisce about your experiences with HyperCard, and that was Percy's kind of introduction to computing, or I guess programming. That was actually my introduction to programing, too, in a weird way.

**KEVIN SCOTT:** Oh, that's awesome.

**CHRISTINA WARREN:** Yeah, and before I – I built webpages, I was building HyperCard things, and what kind of struck me was, as you were talking about how to teach the next generation, and talking about different tooling, the idea of a – or the concept of like a – a HyperCard for AI. That's something that I think would be really, really beneficial. What, what are your thoughts?

**KEVIN SCOTT:** Well, I think he was getting at that a little bit when he was talking about his ideas around program synthesis at the end of the interview. So, and it's really interesting. I find this to be the case with a lot of people that the inspiration, like the thing that first tugged you into computing and programming oftentimes sticks with you, your entire career.

And so he started his computing experience thinking about HyperCard which is this very natural, easy way to express computations, and still, to this day, like the thing that he's most excited about is how you can use these very sophisticated AI machine learning technologies to help people express their needs for compute at a more natural way so that the computer can go help people out. Like I think that's so awesome.

**CHRISTINA WARREN:** Yeah, I do too, and I thought the same thing when he was talking about the program synthesis. That has some people, I think, understandably, maybe freaked out, right, like the idea that, oh, these things can write themselves. But when you put it in that context that it might make things more accessible, and less intimidating, and more available across a variety of different things, I think it becomes really exciting.

**KEVIN SCOTT:** Yeah, I've been saying this a lot lately. There's a way to look at a bunch of machine learning stuff, and get really freaked out about it. And then there's a way to look at machine learning, where you're like, oh, my goodness, this piece of technology is creating a bunch of abundance that didn't exist before, or it's creating opportunity and access that people didn't have before, to more actively participating in the creation of technology. And that's the thing that really excites me about the – the state of machine learning in 2020.

**CHRISTINA WARREN:** Yeah, I agree, I think that there's massive potential for that. And kind of pivoting from that, one of the things that the two of you talked about, towards the end of your conversation was, I guess, the relationship between academia and industry, when it comes to AI and ML. And you were talking about, you know, the tremendous amount of compute that is often needed for these different projects, and for these different research things.

Being someone who's been on both sides, like you have, what do you see as the opportunity for academia and industry to work together, and what do you think are the – what's maybe one of the areas where there's friction, right now?

**KEVIN SCOTT:** Yeah, I think – I think that Percy nailed it in his assessment. So, there's certainly an opportunity for industry to help academia out more with just compute resources. Although, like I think these compute resource constraints, in a sense, aren't the worst thing in the world, like the brutal reality is that, even though it may seem that industry has an abundance of compute relative to a university research lab, if you are inside of a big company, doing these things, the appetite for compute, for these big machine learning projects is so vast that you have scarcity, even inside of big companies.

And so, I think that's a very interesting like constraint for both academia and industry to lean all the way into and to try to figure out clever ways for solving these problems, and I'm super excited about that.

But like the point that he made, which I found particularly interesting is the fact that if we could do a little bit better job sharing our problems with one another, we could probably unlock a ton of creativity that we're not able to bring to bear, solving these problems right now, and that's something that's one of the reason I love doing these podcasts.

So, I'm going to go back and do my job as CTO of Microsoft and see if I can try to make that happen more.

**CHRISTINA WARREN:** I love it.I appreciate you doing that, and I appreciate Percy's work, as well.

Well, that's just about it for us today, but before we end, I just have to say that, Kevin, I have been excitedly anticipating the release of your book, which will be out on April 7th. It's called “*Reprogramming the American Dream*,” and I've actually had a tiny sneak peek, and it's really, really well written. It's really good.

**KEVIN SCOTT:** Oh, thank you. You are too kind. So, I am, I am looking forward to it being out, as well. I got a box of books in the mail, the other day. This is the first book that I've ever written. So, I was – like I had this "pinch me" moment when I opened this box, and there were this stack of hard-cover books that had the words printed in them that I had written, so that's sort of amazing.

**CHRISTINA WARREN:** That's so cool. I love that so much, and I'm definitely going to be recommending it to my friends and my fellow tech nerds out there. Because what I really like about the book is that it really does break down a lot of the things that we've been talking about, in this conversation, like AI, in an understandable way, and a way that is pragmatic, and not – you know, scary.

**KEVIN SCOTT:** Yeah, that was a goal. I was hoping to take a bunch of material that can be relatively complex and present it in a way that, hopefully, it's accessible to a broad audience. So, I think it's actually critically important. Like one of the most important things in AI is to have all of us have a better grounding of what it is, and what it isn't, so that we can make smart decisions about how we want to employ it and how we want to encourage other people to use these technologies on our behalf.

**CHRISTINA WARREN:** I love it. I love it. All right, well, that does it for us. As always, please reach out any time at behindthetech@microsoft.com, tell us what's on your mind, and be sure to tell everyone you know about the show. Thanks for listening.

**KEVIN SCOTT:** See you next time.

[MUSIC]