
Learning Internal Representations by Error Propagation
DAVID E. RUMELHART, GEOFFREY E. HINTON, and RONALD J. WILLIAMS
THE PROBLEM
TABLE 1
TABLE 2
Output Units
Hidden Unit
Input Units
Output Patterns
Internal Representation Units
Input Patterns

ADAM DETOUR

Intelligent Machines

Is AI Riding a One-Trick Pony?

Just about every AI advance you've heard of depends on a breakthrough that's three decades old. Keeping up the pace of progress will require confronting AI's serious limitations.

by James Somers September 29, 2017

This publication from the mid-1980s showed how to train a neural network with many layers. It set the stage for this decade's progress in AI.

I'm standing in what is soon to be the center of the world, or is perhaps just a very large room on the seventh floor of a gleaming tower in downtown Toronto. Showing me around is Jordan Jacobs, who cofounded this place: the nascent Vector Institute, which opens its doors this fall and which is aiming to become the global epicenter of artificial intelligence.

We're in Toronto because Geoffrey Hinton is in Toronto, and Geoffrey Hinton is the father of "deep learning," the technique behind the current excitement about AI. "In 30 years we're going to look back and say Geoff is Einstein—of AI, deep learning, the thing that we're calling AI," Jacobs says. Of the researchers at the top of the field of deep learning, Hinton has more citations than the next three combined. His students and postdocs have gone on to run the AI labs at Apple, Facebook, and OpenAI; Hinton himself is a lead scientist on the Google Brain AI team. In fact, nearly every achievement in the last decade of AI—in translation, speech recognition,

image recognition, and game playing—traces in some way back to Hinton's work.

The Vector Institute, this monument to the ascent of Hinton's ideas, is a research center where companies from around the U.S. and Canada—like Google, and Uber, and Nvidia—will sponsor efforts to commercialize AI technologies. Money has poured in faster than Jacobs could ask for it; two of his cofounders surveyed companies in the Toronto area, and the demand for AI experts ended up being 10 times what Canada produces every year. Vector is in a sense ground zero for the now-worldwide attempt to mobilize around deep learning: to cash in on the technique, to teach it, to refine and apply it. Data centers are being built, towers are being filled with startups, a whole generation of students is going into the field.

This story is part of our November/December 2017 Issue

See the rest of the issue

Subscribe

The impression you get standing on the Vector floor, bare and echoey and about to be filled, is that you're at the beginning of something. But the peculiar thing about deep learning is just how old its key ideas are. Hinton's breakthrough paper, with colleagues David Rumelhart and Ronald Williams, was published in 1986. The paper elaborated on a technique called backpropagation, or backprop for short. Backprop, in the words of Jon Cohen, a computational psychologist at Princeton, is "what all of deep learning is based on—literally everything."

When you boil it down, AI today is deep learning, and deep learning is backprop—which is amazing, considering that backprop is more than 30 years old. It's worth understanding how that happened—how a technique could lie in wait for so long and then cause such an explosion—because once you understand the story of backprop, you'll start to understand the current moment in AI, and in particular the fact that maybe we're not actually at the beginning of a revolution. Maybe we're at the end of one.

Vindication

The walk from the Vector Institute to Hinton's office at Google, where he spends most of his time (he is now an emeritus professor at the University of Toronto), is a kind of living advertisement for the city, at least in the summertime. You can understand why Hinton, who is originally from the U.K., moved here in the 1980s after working at Carnegie Mellon University in Pittsburgh.

When you step outside, even downtown near the financial district, you feel as though you've actually gone into nature. It's the smell, I think: wet loam in the air. Toronto was built on top of forested ravines, and it's said to be "a city within a park"; as it's been urbanized, the local government has set strict restrictions to maintain the tree canopy. As you're flying in, the outer parts of the city look almost cartoonishly lush.

Maybe we're not actually at the beginning of a revolution.

Toronto is the fourth-largest city in North America (after Mexico City, New York, and L.A.), and its most diverse: more than half the population was born outside Canada. You can see that walking around. The crowd in the tech corridor looks less San Francisco—young white guys in hoodies—and more international. There's free health care and good public schools, the people are friendly, and the political order is relatively left-leaning and stable; and this stuff draws people like Hinton, who says he left the U.S. because of the Iran-Contra affair. It's one of the first things we talk about when I go to meet him, just before lunch.

"Most people at CMU thought it was perfectly reasonable for the U.S. to invade Nicaragua," he says. "They somehow thought they owned it." He tells me that he had a big breakthrough recently on a project: "getting a very good junior engineer who's working with me," a woman named Sara Sabour. Sabour is Iranian, and she was refused a visa to work in the United States. Google's Toronto office scooped her up.

Hinton, who is 69 years old, has the kind, lean, English-looking face of the Big Friendly Giant, with a thin mouth, big ears, and a proud nose. He was born in Wimbledon, England, and sounds, when he talks, like the narrator of a children's book about science: curious, engaging, eager to explain things. He's funny, and a bit of a showman. He stands the whole time we talk, because, as it turns out, sitting is too painful. "I sat down in June of 2005 and it was a mistake," he tells me, letting the bizarre line land before explaining that a disc in his back gives him trouble. It means he can't fly, and earlier that day he'd had to bring a contraption that looked like a surfboard to the dentist's office so he could lie on it while having a cracked tooth root examined.

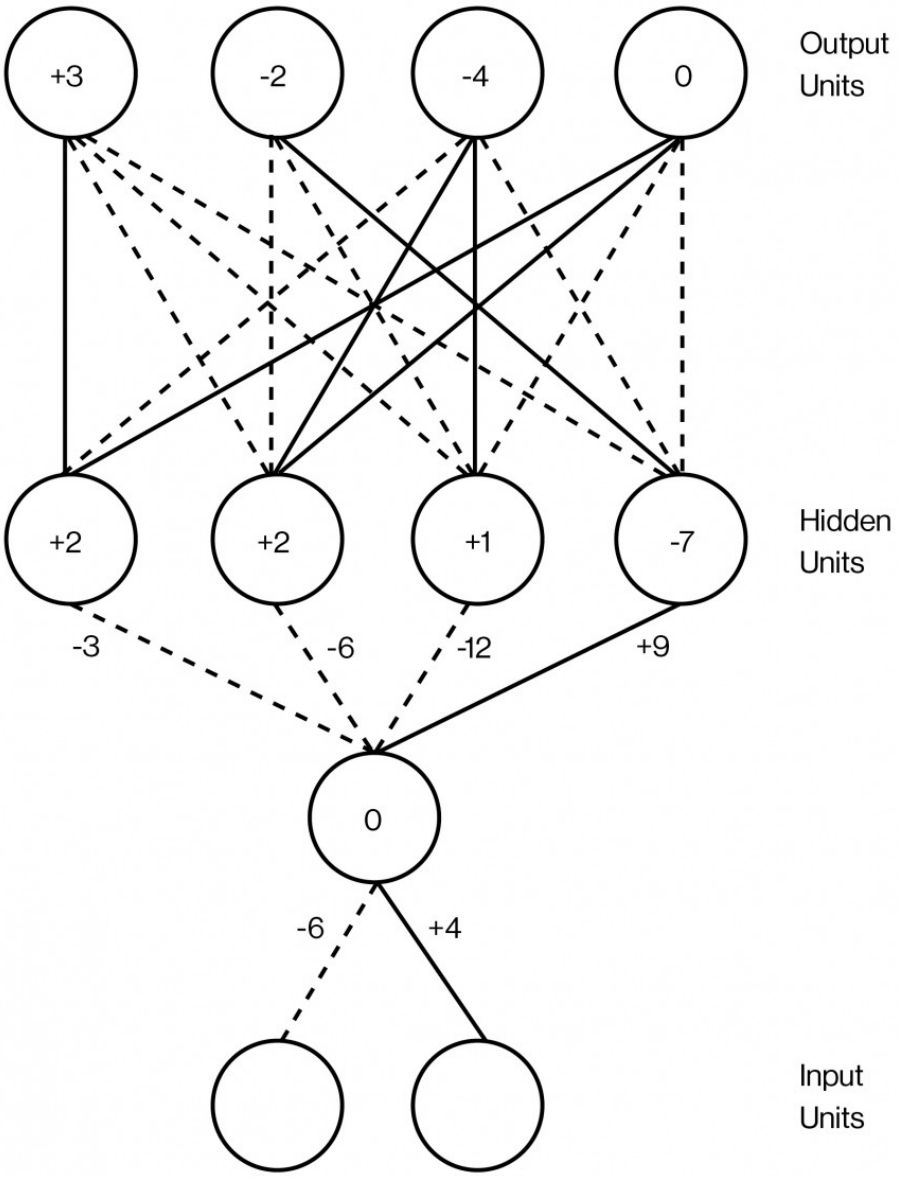
In the 1980s Hinton was, as he is now, an expert on neural networks, a much-simplified model of the network of neurons and synapses in our brains. However, at that time it had been firmly decided that neural networks were a dead end in AI research. Although the earliest neural net, the Perceptron, which began to be developed in the 1950s, had been hailed as a first step toward human-level machine intelligence, a 1969 book by MIT's Marvin Minsky and Seymour Papert, called *Perceptrons*, proved mathematically that such networks could perform only the most basic functions. These networks had just two layers of neurons, an input layer and an output layer. Nets with more layers between the input and output neurons could in theory solve a great variety of problems, but nobody knew how to train them, and so in practice they were useless. Except for a few holdouts like Hinton, *Perceptrons* caused most people to give up on neural nets entirely.

Hinton's breakthrough, in 1986, was to show that backpropagation could train a deep neural net, meaning one with more than two or three layers. But it took another 26 years before increasing computational power made good on the discovery. A 2012 paper by Hinton and two of his Toronto students showed that deep neural nets, trained using backpropagation, beat state-of-the-art systems in image recognition. "Deep learning" took off. To the outside world, AI seemed to wake up overnight. For Hinton, it was a payoff long overdue.

Reality distortion field

A neural net is usually drawn like a club sandwich, with layers stacked one atop the other. The layers contain artificial neurons, which are dumb little computational units that get excited—the way a real neuron gets excited—and pass that excitement on to the other neurons they're connected to. A

neuron's excitement is represented by a number, like 0.13 or 32.39, that says just how excited it is. And there's another crucial number, on each of the connections between two neurons, that determines how much excitement should get passed from one to the other. That number is meant to model the strength of the synapses between neurons in the brain. When the number is higher, it means the connection is stronger, so more of the one's excitement flows to the other.



A diagram from seminal work on "error propagation" by Hinton, David Rumelhart, and Ronald Williams.

One of the most successful applications of deep neural nets is in image recognition—as in the memorable scene in HBO's *Silicon Valley* where the team builds a program that can tell whether there's a hot dog in a picture.

Programs like that actually exist, and they wouldn't have been possible a decade ago. To get them to work, the first step is to get a picture. Let's say, for simplicity, it's a small black-and-white image that's 100 pixels wide and 100 pixels tall. You feed this image to your neural net by setting the excitement of each simulated neuron in the input layer so that it's equal to the brightness of each pixel. That's the bottom layer of the club sandwich: 10,000 neurons (100x100) representing the brightness of every pixel in the image.

You then connect this big layer of neurons to another big layer of neurons above it, say a few thousand, and these in turn to another layer of another few thousand neurons, and so on for a few layers. Finally, in the topmost layer of the sandwich, the output layer, you have just two neurons—one representing "hot dog" and the other representing "not hot dog." The idea is to teach the neural net to excite only the first of those neurons if there's a hot dog in the picture, and only the second if there isn't. Backpropagation—the technique that Hinton has built his career upon—is the method for doing this.

Backprop is remarkably simple, though it works best with huge amounts of data. That's why big data is so important in AI—why Facebook and Google are so hungry for it, and why the Vector Institute decided to set up shop down the street from four of Canada's largest hospitals and develop data partnerships with them.

In this case, the data takes the form of millions of pictures, some with hot dogs and some without; the trick is that these pictures are labeled as to which have hot dogs. When you first create your neural net, the connections between neurons might have random weights—random numbers that say how much excitement to pass along each connection. It's as if the synapses

of the brain haven't been tuned yet. The goal of backprop is to change those weights so that they make the network work: so that when you pass in an image of a hot dog to the lowest layer, the topmost layer's "hot dog" neuron ends up getting excited.

Suppose you take your first training image, and it's a picture of a piano. You convert the pixel intensities of the 100x100 picture into 10,000 numbers, one for each neuron in the bottom layer of the network. As the excitement spreads up the network according to the connection strengths between neurons in adjacent layers, it'll eventually end up in that last layer, the one with the two neurons that say whether there's a hot dog in the picture. Since the picture is of a piano, ideally the "hot dog" neuron should have a zero on it, while the "not hot dog" neuron should have a high number. But let's say it doesn't work out that way. Let's say the network is wrong about this picture. Backprop is a procedure for rejiggering the strength of every connection in the network so as to fix the error for a given training example.

The way it works is that you start with the last two neurons, and figure out just how wrong they were: how much of a difference is there between what the excitement numbers should have been and what they actually were? When that's done, you take a look at each of the connections leading into those neurons—the ones in the next lower layer—and figure out their contribution to the error. You keep doing this until you've gone all the way to the first set of connections, at the very bottom of the network. At that point you know how much each individual connection contributed to the overall error, and in a final step, you change each of the weights in the direction that best reduces the error overall. The technique is called "backpropagation" because you are "propagating" errors back (or down) through the network, starting from the output.

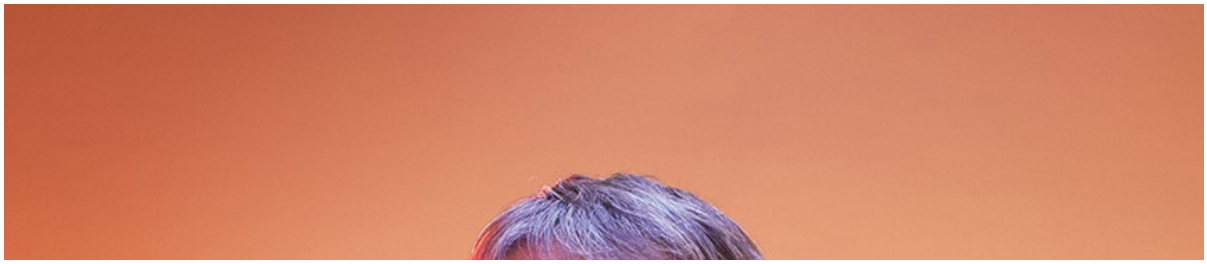
The incredible thing is that when you do this with millions or billions of images, the network starts to get pretty good at saying whether an image has a hot dog in it. And what's even more remarkable is that the individual layers of these image-recognition nets start being able to "see" images in sort of the same way our own visual system does. That is, the first layer might end up detecting edges, in the sense that its neurons get excited when there are edges and don't get excited when there aren't; the layer above that one might be able to detect sets of edges, like corners; the layer above that one might start to see shapes; and the layer above that one might start finding stuff like "open bun" or "closed bun," in the sense of having neurons that respond to either case. The net organizes itself, in other words, into hierarchical layers without ever having been explicitly programmed that way.

A real intelligence doesn't break when you slightly change the problem.

This is the thing that has everybody enthralled. It's not just that neural nets are good at classifying pictures of hot dogs or whatever: they seem able to build representations of ideas. With text you can see this even more clearly. You can feed the text of Wikipedia, many billions of words long, into a simple neural net, training it to spit out, for each word, a big list of numbers that correspond to the excitement of each neuron in a layer. If you think of each of these numbers as a coordinate in a complex space, then essentially what you're doing is finding a point, known in this context as a vector, for each word somewhere in that space. Now, train your network in such a way

that words appearing near one another on Wikipedia pages end up with similar coordinates, and voilà, something crazy happens: words that have similar meanings start showing up near one another in the space. That is, “insane” and “unhinged” will have coordinates close to each other, as will “three” and “seven,” and so on. What’s more, so-called vector arithmetic makes it possible to, say, subtract the vector for “France” from the vector for “Paris,” add the vector for “Italy,” and end up in the neighborhood of “Rome.” It works without anyone telling the network explicitly that Rome is to Italy as Paris is to France.

“It’s amazing,” Hinton says. “It’s shocking.” Neural nets can be thought of as trying to take things—images, words, recordings of someone talking, medical data—and put them into what mathematicians call a high-dimensional vector space, where the closeness or distance of the things reflects some important feature of the actual world. Hinton believes this is what the brain itself does. “If you want to know what a thought is,” he says, “I can express it for you in a string of words. I can say ‘John thought, “Whoops.”’ But if you ask, ‘What is the thought? What does it mean for John to have that thought?’ It’s not that inside his head there’s an opening quote, and a ‘Whoops,’ and a closing quote, or even a cleaned-up version of that. Inside his head there’s some big pattern of neural activity.” Big patterns of neural activity, if you’re a mathematician, can be captured in a vector space, with each neuron’s activity corresponding to a number, and each number to a coordinate of a really big vector. In Hinton’s view, that’s what thought is: a dance of vectors.



Geoffrey Hinton
COURTESY OF GOOGLE

It is no coincidence that Toronto's flagship AI institution was named for this fact. Hinton was the one who came up with the name Vector Institute.

There's a sort of reality distortion field that Hinton creates, an air of certainty and enthusiasm, that gives you the feeling there's nothing that vectors can't do. After all, look at what they've been able to produce already: cars that drive themselves, computers that detect cancer, machines that instantly translate spoken language. And look at this charming British scientist talking about gradient descent in high-dimensional spaces!

It's only when you leave the room that you remember: these "deep learning" systems are still pretty dumb, in spite of how smart they sometimes seem. A computer that sees a picture of a pile of doughnuts piled up on a table and captions it, automatically, as "a pile of doughnuts piled on a table" seems to understand the world; but when that same program sees a picture of a girl brushing her teeth and says "The boy is holding a baseball bat," you realize how thin that understanding really is, if ever it was there at all.

Neural nets are just thoughtless fuzzy pattern recognizers, and as useful as fuzzy pattern recognizers can be—hence the rush to integrate them into just about every kind of software—they represent, at best, a limited brand of intelligence, one that is easily fooled. A deep neural net that recognizes images can be totally stymied when you change a single pixel, or add visual noise that's imperceptible to a human. Indeed, almost as often as we're finding new ways to apply deep learning, we're finding more of its limits. Self-driving cars can fail to navigate conditions they've never seen before. Machines have trouble parsing sentences that demand common-sense understanding of how the world works.

Deep learning in some ways mimics what goes on in the human brain, but only in a shallow way—which perhaps explains why its intelligence can

sometimes seem so shallow. Indeed, backprop wasn't discovered by probing deep into the brain, decoding thought itself; it grew out of models of how animals learn by trial and error in old classical-conditioning experiments. And most of the big leaps that came about as it developed didn't involve some new insight about neuroscience; they were technical improvements, reached by years of mathematics and engineering. What we know about intelligence is nothing against the vastness of what we still don't know.

David Duvenaud, an assistant professor in the same department as Hinton at the University of Toronto, says deep learning has been somewhat like engineering before physics. "Someone writes a paper and says, 'I made this bridge and it stood up!' Another guy has a paper: 'I made this bridge and it fell down—but then I added pillars, and then it stayed up.' Then pillars are a hot new thing. Someone comes up with arches, and it's like, 'Arches are great!'" With physics, he says, "you can actually understand what's going to work and why." Only recently, he says, have we begun to move into that phase of actual understanding with artificial intelligence.

Hinton himself says, "Most conferences consist of making minor variations … as opposed to thinking hard and saying, 'What is it about what we're doing now that's really deficient? What does it have difficulty with? Let's focus on that.'"

It can be hard to appreciate this from the outside, when all you see is one great advance touted after another. But the latest sweep of progress in AI has been less science than engineering, even tinkering. And though we've started to get a better handle on what kinds of changes will improve deep-learning systems, we're still largely in the dark about how those systems work, or whether they could ever add up to something as powerful as the human mind.

It's worth asking whether we've wrung nearly all we can out of backprop. If so, that might mean a plateau for progress in artificial intelligence.

Patience

If you want to see the next big thing, something that could form the basis of machines with a much more flexible intelligence, you should probably check out research that resembles what you would've found had you encountered backprop in the '80s: smart people plugging away on ideas that don't really work yet.

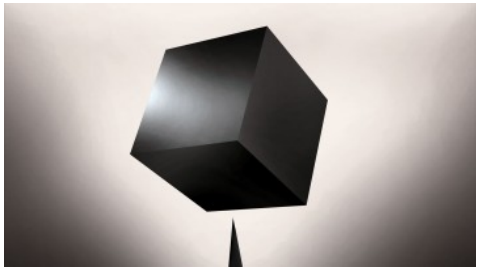
A few months ago I went to the Center for Minds, Brains, and Machines, a multi-institutional effort headquartered at MIT, to watch a friend of mine, Eyal Dechter, defend his dissertation in cognitive science. Just before the talk started, his wife Amy, their dog Ruby, and their daughter Susannah were milling around, wishing him well. On the screen was a picture of Ruby, and next to it one of Susannah as a baby. When Dad asked Susannah to point herself out, she happily slapped a long retractable pointer against her own baby picture. On the way out of the room, she wheeled a toy stroller behind her mom and yelled "Good luck, Daddy!" over her shoulder. "Vámanos!" she said finally. She's two.

"The fact that it doesn't work is just a temporary annoyance."

Eyal started his talk with a beguiling question: How is it that Susannah, after two years of experience, can learn to talk, to play, to follow stories?

What is it about the human brain that makes it learn so well? Will a computer ever be able to learn so quickly and so fluidly?

We make sense of new phenomena in terms of things we already understand. We break a domain down into pieces and learn the pieces. Eyal is a mathematician and computer programmer, and he thinks about tasks—like making a soufflé—as really complex computer programs. But it's not as if you learn to make a soufflé by learning every one of the program's zillion micro-instructions, like "Rotate your elbow 30 degrees, then look down at the countertop, then extend your pointer finger, then …" If you had to do that for every new task, learning would be too hard, and you'd be stuck with what you already know. Instead, we cast the program in terms of high-level steps, like "Whip the egg whites," which are themselves composed of subprograms, like "Crack the eggs" and "Separate out the yolks."



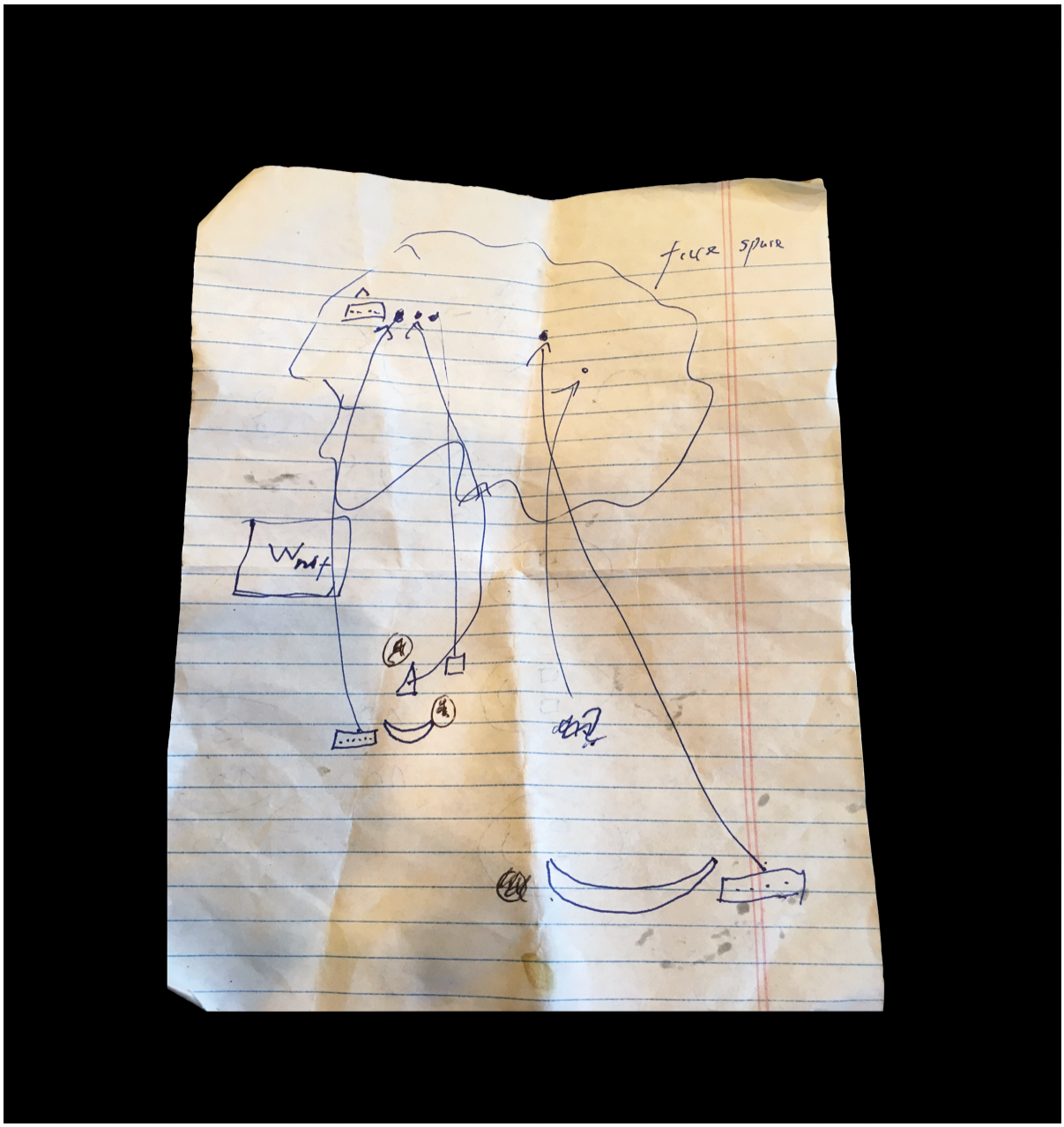
Related Story

The Dark Secret at the Heart of AI
No one really knows how the most advanced algorithms do what they do. That could be a problem.

Computers don't do this, and that is a big part of the reason they're dumb. To get a deep-learning system to recognize a hot dog, you might have to feed it 40 million pictures of hot dogs. To get Susannah to recognize a hot dog, you show her a hot dog. And before long she'll have an understanding of language that goes deeper than recognizing that certain words often appear together. Unlike a computer, she'll have a model in her mind about how the whole world works. "It's sort of incredible to me that people are scared of computers taking jobs," Eyal says. "It's not that computers can't replace

lawyers because lawyers do really complicated things. It's because lawyers read and talk to people. It's not like we're close. We're so far."

A real intelligence doesn't break when you slightly change the requirements of the problem it's trying to solve. And the key part of Eyal's thesis was his demonstration, in principle, of how you might get a computer to work that way: to fluidly apply what it already knows to new tasks, to quickly bootstrap its way from knowing almost nothing about a new domain to being an expert.



Hinton made this sketch for his next big idea, to organize neural nets with "capsules."

Essentially, it is a procedure he calls the “exploration–compression” algorithm. It gets a computer to function somewhat like a programmer who builds up a library of reusable, modular components on the way to building more and more complex programs. Without being told anything about a new domain, the computer tries to structure knowledge about it just by

playing around, consolidating what it's found, and playing around some more, the way a human child does.

His advisor, Joshua Tenenbaum, is one of the most highly cited researchers in AI. Tenenbaum's name came up in half the conversations I had with other scientists. Some of the key people at DeepMind—the team behind AlphaGo, which shocked computer scientists by beating a world champion player in the complex game of Go in 2016—had worked as his postdocs. He's involved with a startup that's trying to give self-driving cars some intuition about basic physics and other drivers' intentions, so they can better anticipate what would happen in a situation they've never seen before, like when a truck jackknifes in front of them or when someone tries to merge very aggressively.

Eyal's thesis doesn't yet translate into those kinds of practical applications, let alone any programs that would make headlines for besting a human. The problems Eyal's working on "are just really, really hard," Tenenbaum said. "It's gonna take many, many generations."

Tenenbaum has long, curly, whitening hair, and when we sat down for coffee he had on a button-down shirt with black slacks. He told me he looks to the story of backprop for inspiration. For decades, backprop was cool math that didn't really accomplish anything. As computers got faster and the engineering got more sophisticated, suddenly it did. He hopes the same thing might happen with his own work and that of his students, "but it might take another couple decades."

As for Hinton, he is convinced that overcoming AI's limitations involves building "a bridge between computer science and biology." Backprop was, in this view, a triumph of biologically inspired computation; the idea initially

came not from engineering but from psychology. So now Hinton is trying to pull off a similar trick.

Neural networks today are made of big flat layers, but in the human neocortex real neurons are arranged not just horizontally into layers but vertically into columns. Hinton thinks he knows what the columns are for—in vision, for instance, they're crucial for our ability to recognize objects even as our viewpoint changes. So he's building an artificial version—he calls them "capsules"—to test the theory. So far, it hasn't panned out; the capsules haven't dramatically improved his nets' performance. But this was the same situation he'd been in with backprop for nearly 30 years.

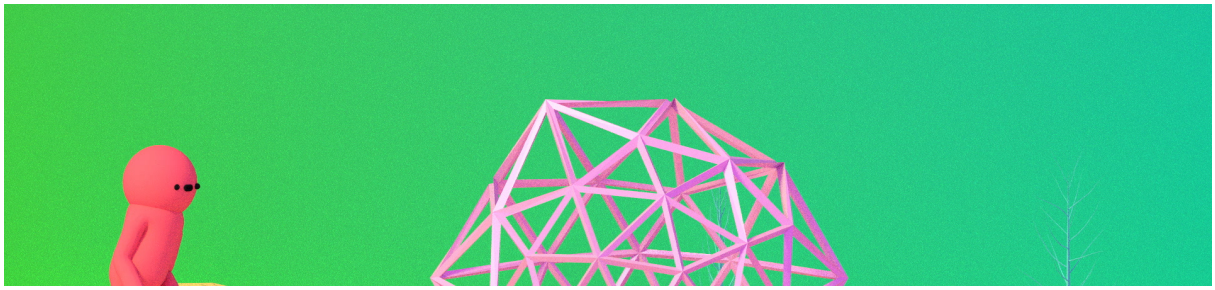
"This thing just has to be right," he says about the capsule theory, laughing at his own boldness. "And the fact that it doesn't work is just a temporary annoyance."

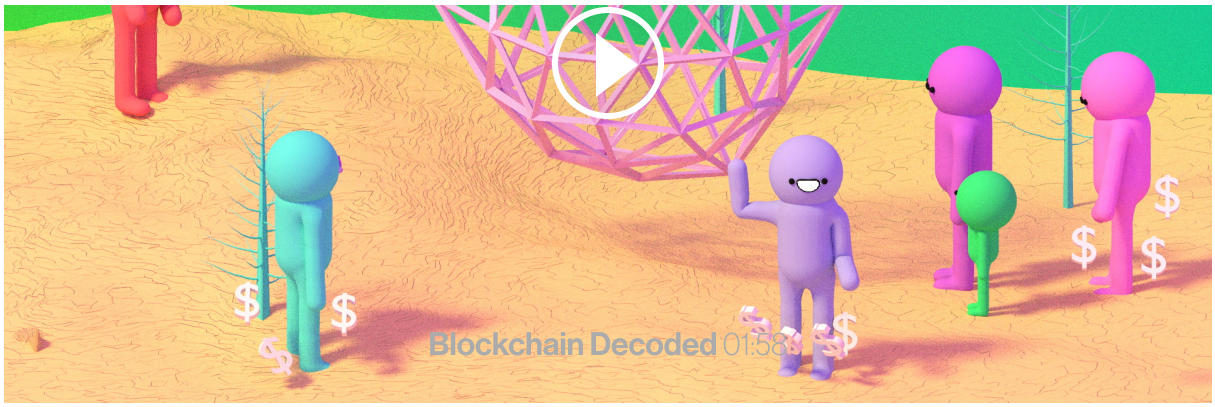
James Somers is a writer and programmer based in New York City. His previous article for MIT Technology Review *was "Toolkits for the Mind" in May/June 2015, which showed how Internet startups are shaped by the programming languages they use.*

Hear more about artificial intelligence at EmTech MIT 2017.
Register now

Related Video

More videos





Recommended for You

01 CRISPR 2.0 Is Here, and It's Way More Precise

02 What Does Work Look Like in 2026? New Statistics Shine Light on Automation's Impacts

03 New Twists in the Road to Quantum Supremacy

04 The Secret Betting Strategy That Beats Online Bookmakers

05 Smartphones Are Weapons of Mass Manipulation, and This Guy Is Declaring War on Them

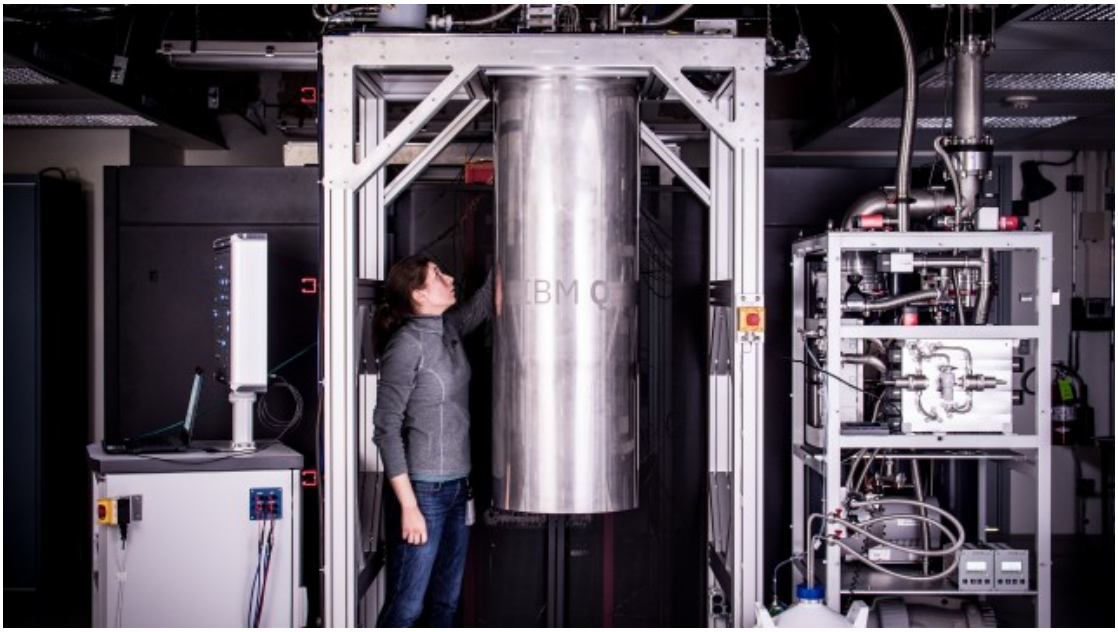
More from Intelligent Machines

Artificial intelligence and robots are transforming how we work and live.

01 **New Twists in the Road to Quantum Supremacy**

Quantum computers will soon surpass conventional ones, but it will take time to make the machines useful.

by Will Knight



02 **Getting To Iconic**

How world-leading brands balance talent and technology for CX excellence

by MIT Technology Custom, International Markets

---

03 **Connectivity and QoL**

How digital consumer habits and ubiquitous technology are driving smart city development in Asia Pacific

by MIT Technology Review Custom, International Markets

More from Intelligent Machines

Want more award-winning journalism?
Subscribe to Insider Plus.

Insider Plus $79.95/year*

Everything included in Insider Basic, plus the digital magazine, extensive archive, ad-free web experience, and discounts to partner offerings and MIT Technology Review events.

Subscribe

See details+

*Prices are for U.S. residents only
See international prices