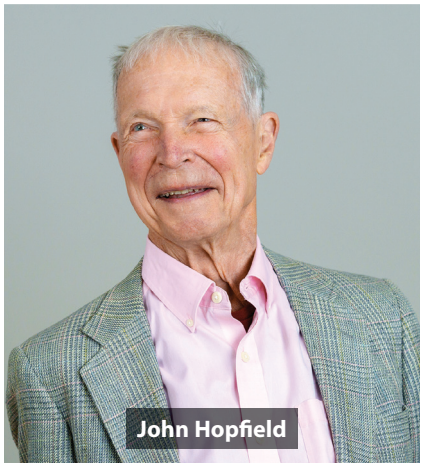
Nobel Prize highlights neural networks' physics roots

The road to the modern machine-learning marvels was paved with ideas from statistical mechanics and collective phenomena.



MATT RASPANTI/PRINCETON UNIVERSITY

JOHNNY GUATTO/UNIVERSITY OF TORONTO

"Garbage in, garbage out." According to the old adage from computer science, what you get from a computer is no better than what you give it. And it would seem to imply that because computers can't think for themselves, they can never do anything more sophisticated than what they've been explicitly instructed to.

But that last part appears to be no longer true. Neural networks—computing architectures, inspired by the human brain, in which signals are passed among nodes called artificial neurons—have, in recent years, been producing wave after wave of stunning results. (See, for example, page 17 of this issue.) Individual artificial neurons perform only the most elementary of computations. But when brought together in large enough numbers, and when fed on enough training data, they acquire capabilities uncannily reminiscent of human intelligence, seemingly out of nowhere.

Physicists are no strangers to the idea of unexpected phenomena emerging from simpler building blocks. A few elementary particles and the rules of their interactions combine to yield almost the whole of the visible world: superconductors, plasmas, and everything in between. Why shouldn't a physics approach to emergent complexity be applied to neural networks too?

Indeed, it was—and still is—as showcased by this year's Nobel Prize in Physics, which goes to Princeton University's John Hopfield and the University of Toronto's Geoffrey Hinton. Beginning in the early 1980s, Hopfield laid the conceptual foundations for physics-based thinking about brain-inspired information processing; Hinton was at the forefront of the decades-long effort to build on those ideas to develop the algorithms used by neural-network models today.

Glassy memory

It was far from obvious, at first, that neural networks would ever grow to be so powerful. As recently as 2011, the flashiest milestones in AI were being achieved by another approach entirely. IBM Watson, the computer that beat Ken Jennings and Brad Rutter at *Jeopardy!*, was not a neural network: It was explicitly programmed with rules for language processing, information retrieval, and logical reasoning. And many researchers thought that was the way to go to create practical AI machines.

In contrast, the early work on neural networks was curiosity-driven research, inspired more by real brains than by computers and their applications. But the nature of the interdisciplinary connection was subtle. "The questions Hopfield addressed are not unrelated to things neuroscientists were worried about," says Princeton's William Bialek. "But this isn't about 'application of physics to X'; rather, it's about introducing a whole point of view that just didn't exist before."

By the 1980s, neuroscientists had known for decades that the brain is composed of neurons, which are connected to one another via synapses and alternate between periods of high and low electrical activity (colloquially, "firing" and "not firing"), and they were studying systems of a few neurons to understand how one neuron's firing affected those it was connected to. "Some thought of neurons in terms of logic gates, like in electronics," says Stanford University's Jay McClelland.

In a landmark 1982 paper, Hopfield took a different approach.[1] In physics, he argued, many important properties of large-scale systems are independent of small-scale details. All materials conduct sound waves, for example, irrespective of exactly how their atoms or molecules interact. Microscopic forces might affect the speed of sound or other acoustic properties, but studying the forces among three or four atoms reveals little about how the concept of sound waves emerges in the first place.

So he wrote down a model of a network of neurons, with an eye more toward computational and mathematical simplicity than neurobiological realism. The model, now known as a Hopfield network, is sketched in figure 1. (The figure shows a five-neuron network for ease of illustration; Hopfield was simulating networks of 30 to 100 neurons.) Each neuron can be in state 1, for firing, or state 0, for not firing. And each neuron was connected to all the others via coupling constants that could have any positive or negative value, depending on whether each synapse favors or disfavors the neurons to both be firing at the same time.

That's exactly the same form as a spin glass, a famously thorny system from condensed-matter physics. (See Physics Today, December 2021, page 17.) Unlike a ferromagnet, in which the couplings are all

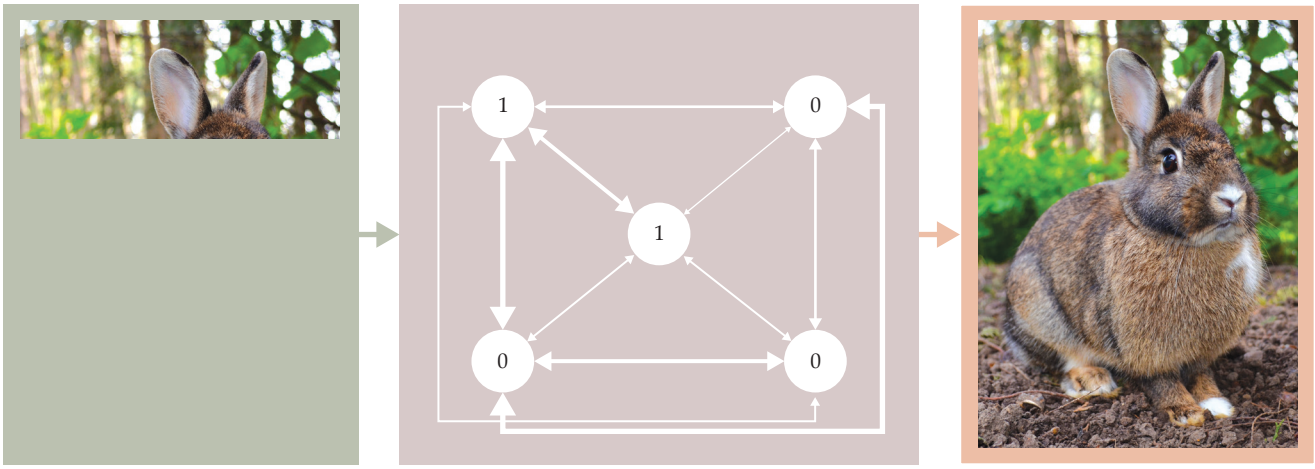


FIGURE 1. A HOPFIELD NETWORK, formally equivalent to a spin glass, functions as an associative memory: When presented with a partially recalled state, it uses an energy-lowering algorithm to fill in the gaps. The memories are stored in the strengths of the connections among the nodes. When John Hopfield showed that with the right combination of connection weights, the network could store many memories simultaneously, he set the stage for physics-based thinking about neural networks. (Figure by Freddie Pagani; rabbit photo by JM Ligero Loarte/Wikimedia Commons/CC BY 3.0.)

positive and the system has a clear ground state with all its spins aligned, a spin glass almost always lacks a state that satisfies all its spins' energetic preferences simultaneously. Its energy landscape is complex, with many local energy minima.

Hopfield argued that the landscape could serve as a memory, with each of the energy-minimizing configurations serving as a state to be remembered. And he presented an elegant way of setting the connection strengths—inspired by what happens at real synapses—so that the memory would store any desired collection of states.

But the Hopfield network is fundamentally different from an ordinary computer memory. In a computer, each item of data to be stored is encoded as a string of ones and zeros in a specific place, and it's recalled by going back to that place and reading out the string. In a Hopfield network, all the items are stored simultaneously in the coupling strengths of the whole network. And they can be recalled associatively, by giving the network a starting point that shares just a few features with one of the remembered states and allowing it to relax to the nearest energy minimum. More often than not, it will recall the desired memory. (See also the articles by Haim Sompolinsky, Physics Today, December 1988, page 70, and John Hopfield, Physics Today, February 1994, page 40.)

Those are both things that happen in real brains. "It was known experimentally in higher animals that brain activity was well spread out, and it involved many neurons," says Hopfield. And associative memory is something you've directly experienced if you've ever recalled a song you've heard before after hearing one random line.

Hopfield's model was a vast simplification of a real brain. Real neurons are intrinsically dynamic, not characterized by static states, and real neuron connections are not symmetric. But in a way, those differences were features, not bugs: They showed that collective, associative memory was an emergent large-scale phenomenon, robust against small-scale details.

Learning how to learn

"Not only is Hopfield a very good physicist, but the Hopfield model is excellent physics by itself," says Leo van Hemmen, of the Technical University of Munich. Still, its 1982 formulation left many intriguing open questions. Hopfield had focused on simulations to show how the system relaxes to an energy minimum; would the model admit a more robust analytical treatment? How many states could the model remember, and what would happen if it was overloaded? Were there better ways of setting the connection strengths than the one Hopfield proposed?

Those questions, and others, were taken on by a flurry of physics-trained researchers who were inspired by Hopfield's work and entered the neural-network field over the 1980s. "Physicists are versatile, curious, and arrogant—in a positive way," says Eytan Domany, of the Weizmann Institute of Science in Israel. "They're willing to study thoroughly and then tackle a problem they've never seen before, if it's interesting. And everyone is excited about understanding the brain."

Another part of the appeal was in how Hopfield had taken a traditional physics problem and turned it on its head. "In most energy-landscape problems, you're given the microscopic interactions, and you ask, What is the ground state? What are the local minima? What is the entire landscape?" says Haim Sompolinsky, of the Hebrew University of Jerusalem. "The 1982 paper did the opposite. We start with the ground states that we want: the memories. And we ask, What are the microscopic interactions that will support those as ground states?"

From there, it was a short conceptual leap to ask, What if the coupling strengths themselves can evolve on their own energy landscape? That is, instead of being preprogrammed with parameters to encode specific memories, can the system improve itself by learning?

Machine learning in neural networks had been tried before. The perceptron—a neural-network-like device that sorted images into simple categories, such as circles and squares—dates back to the 1950s. When provided with a series of training images and a simple algorithm for updating its connections between neurons, it could eventually learn to correctly classify even images it hadn't seen before.

But the perceptron didn't always work: With the way the network was structured, sometimes there wasn't any way of setting the connection strengths

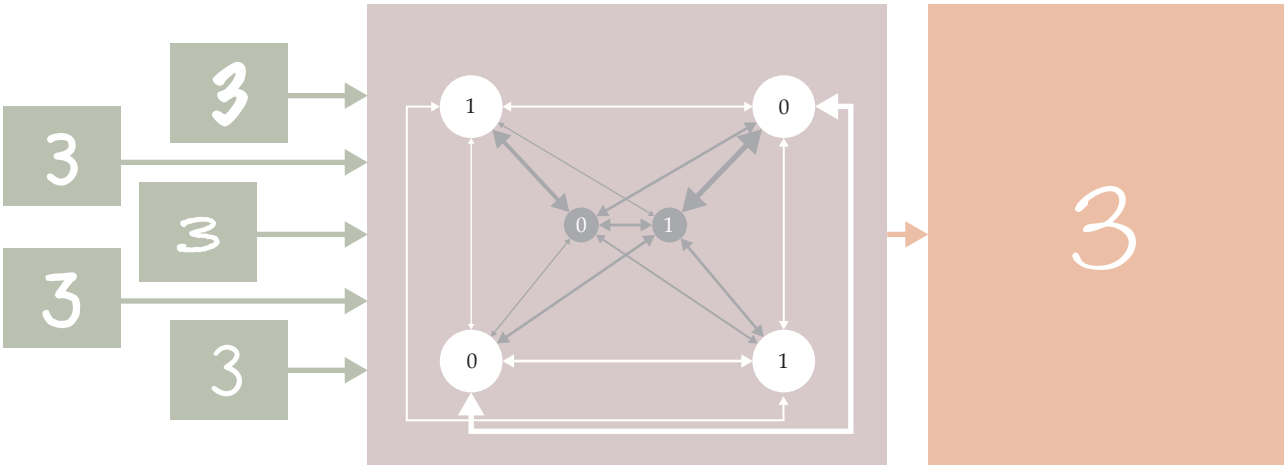


FIGURE 2. A BOLTZMANN MACHINE extends the Hopfield network in two ways: It augments the network to include hidden nodes (shown in the center of the network in gray) that aren't involved in encoding the data, and it operates at a nonzero effective temperature, so that the entire space of configurations can be characterized by a Boltzmann probability distribution. Geoffrey Hinton and colleagues developed a way to train the Boltzmann machine as a generative model: When presented with several inputs that all shared a common feature, it produced more items of the same type. (Figure by Freddie Pagani.)

to perform the desired classification. "When that happened, you could iterate forever, and the algorithm would never converge," says van Hemmen. "That was a big shock." Without a guiding principle to chart a path forward, the field had stalled.

Finding common ground

Hinton didn't come to neural networks from a background in physics. But his collaborator Terrence Sejnowski—who'd earned his PhD under Hopfield in 1978—did. Together, they extended the Hopfield network into something they called the Boltzmann machine, which vastly extended the model's capabilities by explicitly drawing on concepts from statistical physics.[2]

In Hopfield's 1982 simulations, he'd effectively considered the spin-glass network at zero temperature: He allowed the system to evolve its state only in ways that would lower its overall energy. So whatever the starting state, it rolled into a nearby local energy minimum and stayed there.

"Terry and I immediately started thinking about the stochastic version, with nonzero temperature," says Hinton. Instead of a deterministic energy-lowering rule, they used a Monte Carlo algorithm that allowed the system to occasionally jump into a state of higher energy. Given enough time, a stochastic simulation of the network would explore the entire energy landscape, and it would settle into a Boltzmann probability distribution, with all the low-energy states—regardless of whether they're local energy minima—represented with high probability.

"And in 1983, we discovered a really beautiful way to do learning," Hinton says. When the network was supplied with training data, they iteratively updated the connection strengths so that the data states had high probability in the Boltzmann distribution.[3] Moreover, when the input data had something in common—like the images of the numeral 3 in figure 2—then other high-probability states would share the same common features.

The key ingredient for that kind of commonality finding was augmenting the network to include more nodes than just the ones that encode the data. Those hidden nodes, represented in gray in figure 2, allow the system to capture higher-level correlations among the data.

In principle, the Boltzmann machine could be used for machine recognition of handwriting or for distinguishing normal from emergency conditions in a facility such as a power plant. Unfortunately, the Boltzmann machine's learning algorithm is prohibitively slow for most practical applications. It remained a topic of academic research, but it didn't find much real-world use—until it made a surprising reappearance years later.

How the networks work

Around the same time, Hinton was working with cognitive scientist David Rumelhart on another learning algorithm, which would become the secret sauce of almost all of today's neural networks: backpropagation.[4] The algorithm was developed for a different kind of network architecture, called a feedforward network, shown in figure 3. In contrast to the Hopfield network and Boltzmann machine, with their bidirectional connections among nodes, signals in a feedforward network flow in one direction only: from a layer of input neurons, through some number of hidden layers, to the output. A similar architecture had been used in the multilayer perceptron.

Suppose you want to train a feedforward network to classify images. You give it a picture of a rabbit, and you want it to produce the output message "This is a rabbit." But something is wrong, and instead you get the output "This is a turtle." How do you get things back on track? The network might have dozens or hundreds—or today, trillions—of internode connections that contribute to the output, each with its own numerical weight. There's a dizzying number of ways to adjust them all to try to get the output you want.

Backpropagation solves that problem through gradient descent: First, you define an error function that quantifies how far the output you got is from the output you want. Then, calculate the partial derivatives of the error function with respect to each of the internodal weights—a simple matter of repeatedly applying calculus's chain rule. Finally, use those derivatives to adjust the weights in a way that decreases the error.

It might take many repetitions to get

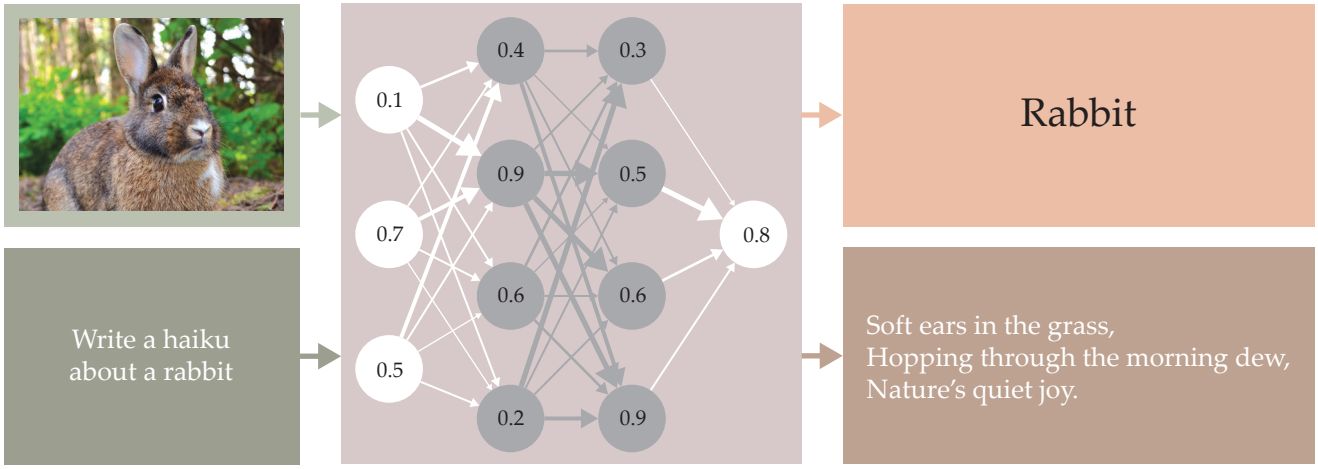


FIGURE 3. A FEEDFORWARD NETWORK, trained by backpropagation, is the basic structure of the neural networks used today. By passing numerical signals from an input layer through hidden layers to an output layer, feedforward networks perform functions that include image classification and text generation. (Figure by Freddie Pagani; rabbit photo by JM Ligero Loarte/Wikimedia Commons/CC BY 3.0; haiku generated by GPT-4, OpenAI, 22 October 2024.)

the error close enough to zero—and you'll want to make sure that the network gives the right output for many inputs, not just one. But those basic steps are used to train all kinds of networks, including proof-of-concept image classifiers and large language models, such as ChatGPT.

Gradient descent is intuitively elegant, and it wasn't conceptually new. "But several elements had to come together to get the backpropagation idea to work," says McClelland. "For one thing, you can't take the derivative of something if it's not differentiable." Real neurons operate more or less in discrete on and off states, and the original Hopfield network, Boltzmann machine, and perceptron were all discrete models. For backpropagation to work, it was necessary to shift to a model in which the node states can take a continuum of values. But those continuous-valued networks had already been introduced, including in a 1984 paper by Hopfield.[5]

A second innovation had to wait for longer. Backpropagation worked well for networks with just a couple of layers. But when the layer count approached five or more—a trifling number by today's standards—some of the partial derivatives were so small that the training took an impractically long time.

In the early 2000s, Hinton found a solution, and it involved his old Boltzmann machine—or rather, a so-called restricted version of it, in which the only connections are those between one hidden neuron and one visible (non-hidden) neuron.[6] Restricted Boltzmann machines (RBMs) are easy to computationally model, because each group of neurons—visible and hidden—could be updated all at once, and the connection weights could all be adjusted together in a single step. Hinton's idea was to isolate pairs of successive layers in a feedforward network, train them as if they were RBMs to get the weights approximately right, and then fine-tune the whole network using backpropagation.

"It was kind of a hacky thing, but it worked, and people got very excited," says Graham Taylor, of the University of Guelph in Canada, who earned his PhD under Hinton in 2009. "It was now possible to train networks with five, six, seven layers. People called them 'deep' networks, and they started using the term 'deep learning.'"

The RBM hack wasn't used for long. Computing power was advancing so quickly—particularly with the realization that graphics processing units (GPUs) were ideally suited to the computations needed for neural networks—that within a few years, it was possible to do backpropagation on even larger networks from a cold start, with no RBMs required.

"If RBM learning hadn't happened, would GPUs have come along anyway?" asks Taylor. "That's arguable. But the excitement around RBMs changed the landscape: It led to the recruitment and training of new students and to new ways of thinking. I think at the very least, it wouldn't have happened the same way."

What's new is old

Today's networks use hundreds or thousands of layers, but their form is little changed from what Hinton described. "I learned about neural networks from books from the 1980s," says Bernhard Mehlig, of the University of Gothenburg in Sweden. "When I started teaching it, I realized that not much is new. It's essentially the old stuff." Mehlig notes that in a textbook he wrote, published in 2021, part 1 of 3 is about Hopfield, and part 2 is about Hinton.

Neural networks now influence a vast number of human endeavors: They're involved in data analysis, web searches, and creating graphics. Are they intelligent? It's easy to dismiss the question out of hand. "There have always been lots of things that machines can do better than humans," says the University of Maryland's Sankar Das Sarma. "That has nothing to do with becoming human. ChatGPT is fabulously good at some things, but at many others, it's not even as good as a two-year-old baby."

An illustrative comparison is the vast data gap between today's neural networks and humans.[7] A literate 20-year-old may have read and heard a few hundred million words in life so far. Large language models, in contrast, are trained on hundreds of billions of words, a number that grows with each new release. When you account for the fact that ChatGPT has the advantage of a thousand times as much life experience as you do, its abilities may seem less like intelligence. But perhaps it doesn't matter if AI fumbles with some tasks if it's good at the right combination of others.

Hinton and Hopfield have both spoken about the dangers of unchecked AI.

Among their arguments is the idea that once machines become capable of breaking up goals into subgoals, they'll quickly deduce that they can make almost any task easier for themselves by consolidating their own power. And because neural networks are often tasked with writing code for other computers, stopping the damage is not as simple as pulling the plug on a single machine.

"There are also imminent risks that we're facing right now," says Mehlig. "There are computer-written texts and fake images that are being used to trick people and influence elections. I think that by talking about computers taking over the world, people take the imminent dangers less seriously."

What can physicists do?

Much of the unease stems from the fact that so little is known about what neural networks are really doing: How do billions of matrix multiplications add up to the ability to find protein structures or write poetry? "People at the big companies are more interested in producing revenue, not understanding," says Das Sarma. "Understanding takes longer. The job of theorists is to understand phenomena, and this is a huge physical phenomenon, waiting to be understood by us. Physicists should be interested in this."

"It's hard not to be excited by what's going on, and it's hard not to notice that we don't understand," says Bialek. "If you want to say that things are emergent, what's the order parameter, and what is it that's emerged? Physics has a way of making that question more precise. Will that approach yield insight? We'll see."

For now, the biggest questions are still overwhelming. "If there were something obvious that came to mind, there would be a horde of people trying to solve it," says Hopfield. "But there isn't a horde of people working on this, because nobody knows where to start."

But a few smaller-scale questions are more tractable. For example, why does backpropagation so reliably reduce the network error to near zero, rather than getting stuck in high-lying local minima like the Hopfield network does? "There was a beautiful piece of work on this a few years ago by Surya Ganguli at Stanford," says Sara Solla, of Northwestern University. "He found that most high-lying minima are really saddle points: It's a minimum in many dimensions, but there's always one in which it's not. So if you keep kicking, you eventually find your way out."

When physics-trained researchers work on problems like that, are they still doing physics? Or have they left physics behind for something else? If "physics" is defined as the study of the natural, physical world, that would arguably exclude artificial neural networks, which by now are wholly human-made abstractions with little resemblance to biological neurons. "We don't build airplanes that flap their wings," says Solla. "And backpropagation is a totally unrealistic mechanism in a real brain. The engineering goal is to make a machine that works. Nature gives us some intuition, but the best solution is not necessarily to copy it."

But must physics be defined solely by its subject matter? "In multidisciplinary fields, what makes the difference between disciplines—mathematics versus computer science versus physics—is their methods and mindsets," says Princeton's Francesca Mignacco. "They're complementary but different. Neural-network models are so complicated that it's hard to achieve rigorous mathematical descriptions. But statistical physics has precisely the tools to tackle the complexity of high-dimensional systems. Personally, I've never stopped asking questions just because they might or might not be physics."

"Physics is limited only by the ingenuity of people applying physical ways of thinking to systems in the real world," says Hopfield. "You can have a narrow view of that, or you can welcome more applied physics. I'm one of the welcomers."

Johanna Miller

References

1. J. J. Hopfield, *Proc. Natl. Acad. Sci. USA* **79**, 2554 (1982).
2. S. E. Fahlman, G. E. Hinton, T. J. Sejnowski, in *Proceedings of the AAAI Conference on Artificial Intelligence, 3*, Association for the Advancement of Artificial Intelligence (1983), p. 109.
3. D. H. Ackley, G. E. Hinton, T. J. Sejnowski, *Cogn. Sci.* **9**, 147 (1985).
4. D. E. Rumelhart, G. E. Hinton, R. J. Williams, *Nature* **323**, 533 (1986).
5. J. J. Hopfield, *Proc. Natl. Acad. Sci. USA* **81**, 3088 (1984).
6. G. E. Hinton, *Neural Comput.* **14**, 1771 (2002); G. E. Hinton, S. Osindero, Y.-W. Teh, *Neural Comput.* **18**, 1527 (2006).
7. M. C. Frank, *Trends Cogn. Sci.* **27**, 990 (2023).