ANNUAL REVIEWS

Annual Review of Condensed Matter Physics

Statistical Mechanics of Deep Learning

Yasaman Bahri,[1] Jonathan Kadmon,[2] Jeffrey Pennington,[1] Sam S. Schoenholz,[1] Jascha Sohl-Dickstein,[1] and Surya Ganguli[1,2]

[1]Google Brain, Google Inc., Mountain View, California 94043, USA

[2]Department of Applied Physics, Stanford University, Stanford, California 94035, USA; email: sganguli@stanford.edu

Annu. Rev. Condens. Matter Phys. 2020. 11:501–28

First published as a Review in Advance on December 9, 2019

The *Annual Review of Condensed Matter Physics* is online at conmatphys.annualreviews.org

https://doi.org/10.1146/annurev-conmatphys-031119-050745

Copyright © 2020 by Annual Reviews. All rights reserved

ANNUAL REVIEWS CONNECT

www.annualreviews.org

- Download figures
- Navigate cited references
- Keyword search
- Explore related articles
- Share via email or social media

Keywords

neural networks, machine learning, dynamical phase transitions, chaos, spin glasses, jamming, random matrix theory, interacting particle systems, nonequilibrium statistical mechanics

Abstract

The recent striking success of deep neural networks in machine learning raises profound questions about the theoretical principles underlying their success. For example, what can such deep networks compute? How can we train them? How does information propagate through them? Why can they generalize? And how can we teach them to imagine? We review recent work in which methods of physical analysis rooted in statistical mechanics have begun to provide conceptual insights into these questions. These insights yield connections between deep learning and diverse physical and mathematical topics, including random landscapes, spin glasses, jamming, dynamical phase transitions, chaos, Riemannian geometry, random matrix theory, free probability, and nonequilibrium statistical mechanics. Indeed, the fields of statistical mechanics and machine learning have long enjoyed a rich history of strongly coupled interactions, and recent advances at the intersection of statistical mechanics and deep learning suggest these interactions will only deepen going forward.

1. INTRODUCTION

Deep neural networks, with multiple hidden layers (1), have achieved remarkable success across many fields, including machine vision (2), speech recognition (3), natural language processing (4), reinforcement learning (5), and even modeling of animals and humans themselves in neuroscience (6, 7), psychology (8, 9), and education (10). However, the methods used to arrive at successful deep neural networks remain a highly practiced art, filled with many heuristics, rather than an exact science. This raises exciting challenges and opportunities for the theoretical sciences in creating a mature theory of deep neural networks that is powerful enough to guide a wide set of engineering design choices in deep learning. Although we are still currently far from any such mature theory, a recently emerged body of work at the intersection of statistical mechanics and deep learning has begun to provide theoretical insights into how deep networks learn and compute, sometimes suggesting new and improved methods for deep learning driven by these theoretical insights.

Here, we review this body of work, which builds upon a long and rich history of interaction between statistical mechanics and machine learning (11–15). Interestingly, this body of work leads to many new bridges between statistical mechanics and deep learning as we discuss below. In the remainder of this introduction, we provide frameworks for two major branches of machine learning. The first is supervised learning, which concerns the process of learning input–output maps from examples. The second is unsupervised learning, which concerns the process of learning and exploiting hidden patterns of structure in data. With these two frameworks in hand, we introduce in Section 1.3 several foundational theoretical questions of deep learning discussed in this review, and their connections to a diversity of topics related to statistical mechanics.

1.1. Overall Framework of Supervised Learning

Image classification is a classic example of supervised learning. In the image classification problem, one must learn a mapping from a pixel representation of an image to a class label for that image (e.g., cat, dog). To learn this map, a neural network is trained on a training set of images along with their correct class label. This is called a supervised learning problem because the correct class labels are given to the network during training. Indeed, a seminal advance that popularized deep learning was a significant improvement in image classification by deep networks (2).

More formally, the simplest version of a feed-forward neural network with D layers is specified by D weight matrices $\mathbf{W}^1, \ldots, \mathbf{W}^D$ and D layers of neural activity vectors $\mathbf{x}^1, \ldots, \mathbf{x}^D$, with N_l neurons in each layer l, so that $\mathbf{x}^l \in \mathbb{R}^{N_l}$ and $\mathbf{W}^l$ is an $N_l \times N_{l-1}$ matrix. The feed-forward dynamics elicited by an input $\mathbf{x}^0$ presented to the network is given by

$$\mathbf{x}^l = \phi(\mathbf{h}^l), \qquad \mathbf{h}^l = \mathbf{W}^l \mathbf{x}^{l-1} + \mathbf{b}^l \quad \text{for } l = 1, \ldots, D, \tag{1.}$$

where $\mathbf{b}^l$ is a vector of biases, $\mathbf{h}^l$ is the pattern of inputs to neurons at layer l, and ϕ is a single neuron scalar nonlinearity that acts component-wise to transform inputs $\mathbf{h}^l$ to activities $\mathbf{x}^l$. We henceforth collectively denote all N neural network parameters $\{\mathbf{W}^l, \mathbf{b}^l\}_{l=1}^D$ by the N-dimensional parameter vector $\mathbf{w}$, and the final output of the network in response to the input $\mathbf{x}^0$ by the vector $\mathbf{y} = \mathbf{x}^D(\mathbf{x}^0, \mathbf{w})$, where the function $\mathbf{x}^D$ is defined recursively in Equation 1.

A supervised learning task is specified by a joint distribution $\mathcal{P}(\mathbf{x}^0, \mathbf{y})$ over possible inputs $\mathbf{x}^0$ and outputs $\mathbf{y}$. A key goal of supervised learning is to find an optimal set of parameters that minimizes the test error on a randomly chosen input–output pair $(\mathbf{x}^0, \mathbf{y})$:

$$\mathcal{E}_{\text{Test}}(\mathbf{w}) = \int d\mathbf{x}^0 \, d\mathbf{y} \, \mathcal{P}(\mathbf{x}^0, \mathbf{y}) \mathcal{L}(\mathbf{y}, \hat{\mathbf{y}}), \tag{2.}$$

where the loss function $\mathcal{L}(\mathbf{y}, \hat{\mathbf{y}})$ penalizes any discrepancy between the correct output $\mathbf{y}$ and the network prediction $\hat{\mathbf{y}} = \mathbf{x}^D(\mathbf{x}^0, \mathbf{w})$. For example, a simple loss function is the squared loss $\mathcal{L} = \frac{1}{2}(\mathbf{y} - \hat{\mathbf{y}})^2$. However, in real-world applications, it may not be possible to either directly access or even mathematically specify the data distribution $\mathcal{P}$. For example, in image classification, $\mathbf{x}^0$ could denote a vector of pixel intensities of an image, whereas $\mathbf{y}$ could denote a probability distribution over image category labels. However, one can often access a finite data set $\mathcal{D} = \{\mathbf{x}^{0,\mu}, \mathbf{y}^\mu\}_{\mu=1}^P$ of P independent identically distributed (i.i.d.) samples drawn from $\mathcal{P}$ (e.g., example images of cats and dogs). One can then attempt to choose parameters $\mathbf{w}$ to minimize the training error,

$$\mathcal{E}_{\text{Train}}(\mathbf{w}, \mathcal{D}) = \frac{1}{P} \sum_{\mu=1}^{P} \mathcal{L}(\mathbf{y}^\mu, \hat{\mathbf{y}}^\mu), \qquad 3.$$

or the average mismatch between correct answers $\mathbf{y}^\mu$ and network predictions $\hat{\mathbf{y}}^\mu = \mathbf{x}^D(\mathbf{x}^{0,\mu}, \mathbf{w})$ on the specific training set $\mathcal{D}$. Many approaches to supervised learning attempt to minimize this training error, potentially with an additional cost function on $\mathbf{w}$ to promote generalization to accurate predictions on new inputs, as we discuss below.

1.2. Overall Framework of Unsupervised Learning

In addition to learning input–output maps, another key branch of machine learning, known as unsupervised learning, concerns modeling and understanding the structure of complex data. For example, how can we describe the structure of natural images, sounds, and language? If we could accurately model probability distributions over such complex data, then we could generate naturalistic data, as well as correct errors in image acquisition (16), speech recordings (17), or human language generation (4).

Of course, the distribution over such complex data as images and sounds cannot be mathematically specified, but we often have access to an empirical distribution of P samples:

$$q(\mathbf{x}) = \frac{1}{P} \sum_{\mu=1}^{P} \delta(\mathbf{x} - \mathbf{x}^\mu). \qquad 4.$$

For example, each $\mathbf{x}^\mu$ could denote a vector of pixel intensities for images, or a time series of pressure variations for sound.

The goal of unsupervised learning is to adjust the parameters $\mathbf{w}$ of a family of distributions $p(\mathbf{x}; \mathbf{w})$ to find one similar to the data distribution $q(\mathbf{x})$. This is often done by maximizing the log likelihood of the data with respect to model parameters $\mathbf{w}$:

$$l(\mathbf{w}) = \int d\mathbf{x}\, q(\mathbf{x}) \log p(\mathbf{x};\ \mathbf{w}). \qquad 5.$$

This learning principle modifies p to assign high probability to data points, and consequently low probability elsewhere, thereby moving the model distribution, $p(\mathbf{x}; \mathbf{w})$, closer to the data distribution, $q(\mathbf{x})$. Indeed, we review further connections between the log-likelihood function and an information theoretic divergence between distributions, as well as free energy and entropy, in Section 6.2.

Once a good model $p(\mathbf{x}; \mathbf{w})$ is found, it has many uses. For example, one can sample from it to imagine new data. One can also use it to denoise or fill in missing entries in a data vector $\mathbf{x}$. Furthermore, if the distribution p consists of a generative process that transforms latent, or hidden, variables $\mathbf{h}$ into the visible data vector $\mathbf{x}$, then the inferred latent variables $\mathbf{h}$, rather than the

data vector itself, can aid in solving subsequent supervised learning tasks. This approach has been very successful, for example, in natural language processing, where the hidden layers of a network trained simply to generate language form useful internal representations for solving subsequent language processing tasks (4).

Interestingly, the process of choosing p can be thought of as an inverse statistical mechanics problem (18). Traditionally, many problems in the theory of equilibrium statistical mechanics involve starting from a Boltzmann distribution $p(\mathbf{x}; \mathbf{w})$ over microstates $\mathbf{x}$, with couplings $\mathbf{w}$, and computing bulk statistics of $\mathbf{x}$ from p. In contrast, machine learning involves sampling from microstates $\mathbf{x}$ and deducing an appropriate distribution $p(\mathbf{x}; \mathbf{w})$.

1.3. Foundational Theoretical Questions in Deep Learning

With the above minimal frameworks for supervised and unsupervised learning in hand, we can now introduce foundational theoretical questions in the field of deep learning and how ideas from statistical physics have begun to shed light on these questions. On the supervised side, we discuss four questions. First, what is the advantage of depth D? In principle, what functions can be computed in Equation 1 for large, but not small, D? We address this question in Section 2 by making a connection to dynamical phase transitions between order and chaos.

Second, many methods for minimizing the training error in Equation 3 involve descending the error landscape over the parameter vector $\mathbf{w}$ given by $\mathcal{E}_{\text{Train}}(\mathbf{w}, \mathcal{D})$ via (stochastic) gradient descent. What is the shape of this landscape and when can we descend to points of low training error? We address these questions in Section 3, making various connections to the statistical mechanics of energy landscapes with quenched disorder, including phenomena like random Gaussian landscapes, spin glasses, and jamming. Indeed $\mathcal{E}_{\text{Train}}(\mathbf{w}, \mathcal{D})$ could be thought of as such an energy function over thermal degrees of freedom $\mathbf{w}$, where the data $\mathcal{D}$ play the role of quenched disorder.

Third, when minimizing $\mathcal{E}_{\text{Train}}(\mathbf{w}, \mathcal{D})$ via gradient descent, one must start at an initial point $\mathbf{w}$, which is often chosen randomly. How can one choose the random initialization to accelerate subsequent gradient descent? In Section 4, we show that theories of signal propagation through random deep networks provide clues to good initializations, making connections to topics in random matrix theory, free probability, and functional path integrals.

Fourth, though many learning algorithms minimize $\mathcal{E}_{\text{Train}}$ in Equation 3, possibly with extra regularization on the parameters $\mathbf{w}$, the key goal is to minimize the inaccessible test error $\mathcal{E}_{\text{Test}}$ in Equation 2 on a randomly chosen new input not necessarily present in the training data $\mathcal{D}$. It is then critical to achieve a small generalization error $\mathcal{E}_{\text{Gen}} = \mathcal{E}_{\text{Test}} - \mathcal{E}_{\text{Train}}$. When can one achieve such a small generalization error, especially in situations in which the number of parameters N can far exceed the number of data points P? We address this question in Section 5, making connections to topics like phase transitions in random matrix spectra, free field theories, and interacting particle systems.

On the unsupervised side, the theoretical development is much less mature. However, in Section 6, we review work in deep unsupervised learning that connects to ideas in equilibrium statistical mechanics, like free-energy minimization, as well as nonequilibrium statistical mechanics, like the Jarzynski equality and heat dissipation in irreversible processes.

2. EXPRESSIVITY OF DEEP NETWORKS

Seminal results (19, 20) demonstrate that shallow networks, with only one hidden layer of neurons, can universally approximate any Borel measurable function from one finite-dimensional space to another, given enough hidden neurons. These results then raise a fundamental question: Why are

deeper neural networks with many hidden layers at all functionally advantageous in solving key problems in machine learning and artificial intelligence?

2.1. Efficient Computation of Special Functions by Deep Networks

Importantly, the early results on function approximation in References 19 and 20 do not specify how many hidden neurons are required to approximate, or express, any given function by a shallow network. A key factor thought to underlie the success of deep networks over their shallow cousins is their high expressivity. This notion corresponds primarily to two intuitions. The first is that deep networks can compactly express highly complex functions over input space in a way that shallow networks with one hidden layer and the same number of neurons cannot. The second intuition, which has captured the imagination of machine learning (21) and neuroscience (22) alike, is that deep neural networks can disentangle highly curved decision boundaries in input space into flattened decision boundaries in hidden space, to aid the performance of simple linear classifiers. To more precisely define a decision boundary, consider the deep network $\mathbf{y} = \mathbf{x}^D(\mathbf{x}^0, \mathbf{w})$ in Equation 1, where the final output $\mathbf{y}$ is restricted to a scalar function y. This network can perform a binary classification task by partitioning the input vectors $\mathbf{x}^0$ into two classes, according to whether $y = \mathbf{x}^D(\mathbf{x}^0, \mathbf{w})$ is positive or negative. The codimension 1 manifold obeying the equation $\mathbf{x}^D(\mathbf{x}^0, \mathbf{w}) = 0$ is then the network's decision boundary. Note that one could also define the decision boundary similarly in the penultimate hidden layer $\mathbf{x}^{D-1}$. Whereas the decision boundary in this hidden layer must be a linear hyperplane, by virtue of the linear map from $\mathbf{x}^{D-1}$ to the scalar b^D, the decision boundary in input space could potentially be highly curved by virtue of the highly nonlinear map from $\mathbf{x}^0$ to $\mathbf{x}^{D-1}$ in Equation 1.

Focusing on the first intuition, several works have exhibited examples of particular complex functions that can be computed with a number of neurons that grows polynomially with the number of input dimensions when using a deep network, but require a number of neurons that instead grows exponentially in the input dimension when using a shallow network (23–27). The theoretical techniques employed in these works both limited the applicability of theory to specific nonlinearities and dictated the particular measure of deep functional complexity involved. For example, Reference 23 focused on rectified linear unit (ReLU) nonlinearities and number of linear regions as a complexity measure; Reference 24 focused on sum-product networks, which compute polynomials, and the number of monomials in the polynomial as complexity measure; and Reference 28 focused on Pfaffian nonlinearities and topological measures of complexity, like the sum of Betti numbers of a decision boundary. These works thus left open a fundamental question: Are the particular example functions efficiently computed by particular deep networks merely rare curiosities, or in some sense is any function computed by a generic deep network, with more general nonlinearities, not efficiently computable by a shallow network?

2.2. Expressivity Through Transient Chaos

Recent work (29) addressed this question by combining Riemannian geometry and dynamical mean-field theory (30) to analyze the propagation of signals through random deep networks in which the weights and biases are chosen i.i.d. from zero-mean, Gaussian distributions. In a phase plane formed by the variance of the weights and biases, this work revealed a dynamical phase transition between ordered and chaotic regimes of signal propagation [see **Figure 1*a*,*b*** for an example where the nonlinearity ϕ in Equation 1 is taken to be $\phi(x) = \tanh x$]. Intuitively, for small weights, relative to the strength of biases, nearby input points coalesce as they propagate through the layers of a deep network and the feed-forward map stays within the linear regime. However, for large weights, signal propagation corresponds to alternating linear expansion and nonlinear

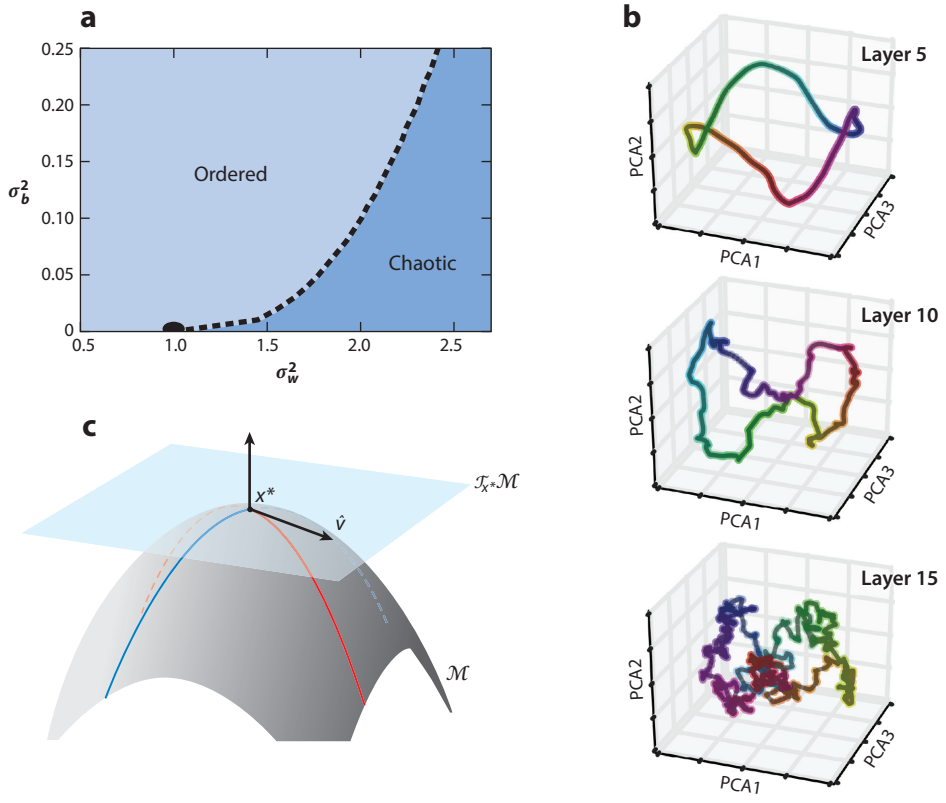

Figure 1

Deep neural expressivity through transient chaos. (*a*) A dynamical phase transition between ordered and chaotic signal propagation in random deep neural networks as a function of the weight variance σ_w^2 and bias variance σ_b^2 (29, 31). Such a phase transition holds for any smooth, odd saturating nonlinearity with finite slope at the origin. Results are shown for $\phi(x) = \tanh x$. (*b*) An example of the propagation of a simple manifold through multiple layers in the chaotic regime. (*c*) An example of a decision boundary or a codimension 1 manifold $\mathcal{M}$ in input space. Eigenvalues associated with the diagonalization of a normalized quadratic approximation to the manifold at a point x^* yields principal curvatures of $\mathcal{M}$ at x^* quantifying departures away from the tangent plane $\mathcal{T}_{x^*}\mathcal{M}$. A dynamic mean-field theory for the propagation of these principal curvatures was developed in Reference 29. This theory revealed that the principal curvatures of decision boundaries in input space associated with flat boundaries in output space grow exponentially with depth in the chaotic regime in Reference 29. Abbreviation: PCA, principal components analysis.

folding, leading to an exponential divergence of nearby inputs without the norm of inputs blowing up, just like chaotic propagation in recurrent dynamical systems with positive Lyapunov exponents. We review this phase transition in more detail in Section 4.

In this chaotic regime, global measures of the length and integrated extrinsic curvature of simple one-dimensional input manifolds typically grow exponentially with depth for random networks (29; **Figure 1*b***), whereas the corresponding measure of length can grow at most as the square root of the width in a shallow network, no matter how one chooses the weights. This then demonstrates the result that random deep networks cannot be approximated by shallow networks unless they have exponentially many more neurons. Furthermore, in this chaotic regime, flat decision boundaries in the output space correspond to decision boundaries in input space with

principal extrinsic curvatures that grow exponentially with depth (29; **Figure 1*c***). Furthermore, a closely related study focusing on the popular ReLU nonlinearity demonstrated also that the length of input manifolds grew exponentially with depth, that neural network training was more sensitive to the lower layers, and intriguingly that the length of trajectories could be a beneficial regularizer (32).

In general, much more theoretical work is needed to understand how and when deep networks can efficiently express more natural functions over natural input domains of the type we would like to learn in artificial intelligence. Interesting early directions along these lines include an analysis of general classes of compositional functions (33), as well as explorations of the capacity of simple neural networks to classify smooth manifolds (34).

3. THE ERROR LANDSCAPE OF NEURAL NETWORKS

Even if a deep network can express a desired function for some choice of parameters, it is unclear when one can successfully find this set of parameters by descending the training error $\mathcal{E}_{\text{Train}}(\mathbf{w}, \mathcal{D})$ in Equation 3 via (stochastic) gradient descent. Typical properties of the shape of this error landscape, its dependence on the number of training examples and network architecture, and its implications for the learning dynamics then become questions of intense interest. In this section, we review insights gained from various analogies that have been drawn between neural network error landscapes and complex energy landscapes in statistical mechanics, as well as insights gained from controlled numerical explorations of neural network error landscapes.

3.1. Random Gaussian Landscapes and Saddles

In machine learning, much early work was motivated by powerful theoretical guarantees afforded by optimization over convex landscapes, in which every local minimum is a global minimum (35). In contrast, optimization over nonconvex landscapes was treated with suspicion, as conventional wisdom suggested such landscapes may be riddled with local minima at high error, which could trap gradient-descent dynamics and impede performance. Although generic nonconvex functions over a small number of variables may indeed have such high error local minima, this is typically not the case in higher dimensions.

Classic work in the statistical physics of smooth random Gaussian landscapes over N variables revealed a very different picture for large N (36, 37). Such a random Gaussian function $E(\mathbf{w})$ for $\mathbf{w} \in \mathbb{R}^N$ is characterized by the fact that its function values $E(\mathbf{w}^\alpha)$ at any finite set of points $\mathbf{w}^\alpha$ for $\alpha = 1, \ldots, M$ are jointly Gaussian distributed with zero mean and a covariance matrix $K_{\alpha\beta} = \mathcal{K}(||\mathbf{w}^\alpha - \mathbf{w}^\beta||^2/N)$. Here, the kernel function $\mathcal{K}(\Delta)$, which determines the correlation of function values at pairs of points separated by a normalized squared distance Δ, decays with increasing Δ. Thus, this ensemble represents a null model of functions over high-dimensional spaces with no assumed structure other than a notion of locality in which nearby function values are similar.

The statistics of critical points of this null ensemble exhibit an interesting typical structure, providing a window into the shape of generic functions over high-dimensional spaces (36). In particular, any critical point $\mathbf{w}$ at which the gradient vanishes [i.e., $\partial_i E(\mathbf{w}) = 0$ for all $i = 1, \ldots, N$] can be characterized by two features: (*a*) the height $E(\mathbf{w})$ of the critical point, and (*b*) the index or fraction, f, of directions in which the function curves down. This latter fraction f is defined to be the fraction of negative eigenvalues of the Hessian matrix $H_{ij} = \frac{\partial^2 E}{\partial w_i \partial w_j}$. The results of Reference 36 found a strong correlation between E and f: The higher a critical point, the larger the number of negative curvature directions. This automatically implies that at high error E, local minima with $f = 0$ would be exponentially rare relative to saddle points in which f is nonzero. Intuitively, the

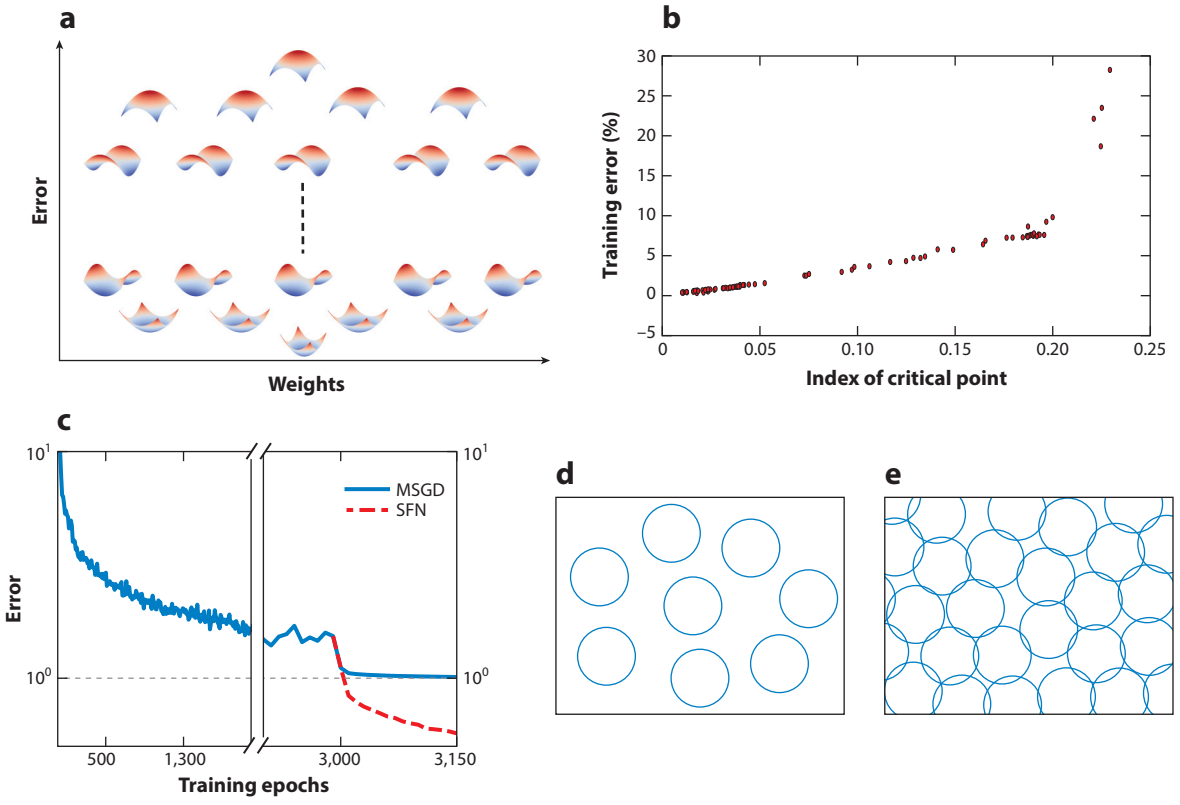

Figure 2

Analogies for the error landscape of neural networks. (*a*) A schematic picture of the typical structure of critical points of smooth random Gaussian landscapes. Critical points with more negative directions occur at higher levels. (*b*) A strong correlation between E and f across critical points on the error landscape of a neural network used in practice (38). (*c*) Saddle-free Newton (SFN), a particular algorithm developed in Reference 38, can rapidly escape saddle points in cases in which stochastic gradient descent slows down. (*d*) Particles in an unjammed state of zero-energy density. (*e*) Particles in a jammed state of positive energy density. Abbreviation: MSGD, momentum stochastic gradient descent.

chances that a generic function E curves up in all N dimensions for large N is exponentially small in N, unless you are already near the bottom of the function. A schematic view of this picture is shown in **Figure 2*a***.

Although the correlation between E and f was computed specifically for random Gaussian landscapes, early work (38) conjectured that this correlation may be more universally true of generic functions over high-dimensional spaces, including error functions of neural networks. A numerical test of this conjecture, by using Newton's method to search for saddle points of any index, confirmed a strong correlation between error level E and index f for all such critical points found using this method (**Figure 2*b***). Furthermore, these researchers (38) developed an algorithm to rapidly escape saddle points in situations in which the dynamics of stochastic gradient descent (SGD) suggested one might have been stuck in an illusory local minimum (**Figure 2*c***).

These physics-based conclusions regarding the absence of high-error local minima for large neural networks are consistent with more mathematical work proving the nonexistence of such minima in simpler settings. Reference 39 proved that the error landscape of linear neural networks

with one hidden layer has no local minima that are not also global minima; all higher-error critical points are saddle points. Furthermore, Reference 40 extended this result to deeper linear networks.

3.2. An Analogy to Spin Glasses

Recent work (41) also exhibited a connection between the error landscape of neural networks and the energy function of a well-studied model in statistical physics, the spherical spin glass. In general, the error landscape of neural networks is a complex function over the synaptic weights $\mathbf{w}$ that also depends on the training data $\mathcal{D}$. A sequence of approximations and simplifying assumptions about both the data and the dependence of the error function over the weights leads to the following toy model of a neural network error landscape:

$$E(\mathbf{w}) = \frac{1}{N^{(D-1)/2}} \sum_{i_1,\ldots,i_D}^{N} X_{i_1,\ldots,i_D} w_{i_1}, w_{i_2}, \ldots, w_{i_D}. \qquad 6.$$

Here, $X_{i_1,\ldots,i_D}$ are random numbers reflecting random simplified data $\mathcal{D}$, w_i are components of the vector of N synaptic weights of the neural network, and D is the depth of the network. Additionally, the weights are assumed to obey a spherical constraint $\mathbf{w} \cdot \mathbf{w} = N$. This error function corresponds to the energy function of the well-known D-spin spherical spin glass (42, 43). The simplifications and assumptions in Reference 41 that lead from neural network error landscapes to Equation 6 are quite strong and unlikely to hold in practice, so Equation 6 should indeed be thought of as a toy model. Nevertheless, one may hope that typical features of the energy function of this toy model may be similar to those of the error landscape of neural networks.

A rigorous analysis (44, 45) of the shape of the error function in Equation 6 revealed an interesting structure for the critical points. Indeed, $E(\mathbf{w})$ can be thought of as a random Gaussian function on the sphere with a specific correlation kernel, and so its critical points qualitatively behave as they do in **Figure 2*a***. In particular, a typical critical point with a certain fraction f of negative curvature directions is most likely to be found within a narrow band of error levels, with the height of the band increasing with f. This picture was confirmed numerically in Reference 41 for deep neural networks trained on the Modified National Institute of Standards and Technology (MNIST) database of digit classification examples.

More recent work (46) undertook a careful comparison of the dynamics of SGD on neural networks versus the D-spin spherical spin glass energy function, finding interesting commonalities but also differences. In mean-field glassy systems, both physical (47) and rigorous (48) methods indicate that gradient-descent dynamics converge without barrier crossing to the widest and the highest minima, despite the existence of deeper local and global minima. In contrast, the work of Reference 46 found additional interesting aging phenomena in gradient-descent dynamics that were indicative of the prevalence of more flat directions as one descends the training error.

3.3. An Analogy to Jamming

By considering a particular loss function $\mathcal{L}$ known as the hinge loss, the authors of References 49 and 50 found an interesting analogy between jamming (51) and the error landscape of deep neural networks, building on a prior such analogy for the perceptron (52). The hinge loss is often used in classification settings in which the neural network output is a single real number whose sign indicates one of two classes. The hinge loss as a function of weight space $\mathbf{w}$ then distinguishes each of the P training examples as either satisfied (i.e., classified with the correct sign with a threshold margin) or unsatisfied. Each point $\mathbf{w}$ in N-dimensional network parameter space then yields a fraction of unsatisfied examples, and ideally training should adjust parameters $\mathbf{w}$ to reduce this fraction.

We now describe the problem of jamming in a parallel notation to bring out the analogy with neural network training. A simple version of the jamming problem considers a set of K hard spheres in a fixed volume (**Figure 2*d*,*e***). Let the vector $\mathbf{w}$ parameterize the N dimensional configuration space of all K sphere positions. The N degrees of freedom associated with all particle positions are analogous to the N degrees of freedom associated with all neural network parameters. Now each of the $P = \binom{K}{2}$ pairwise distances between particles contributes an energy to the particle configuration that is positive if the particle pair overlaps, and zero otherwise. In the analogy to neural networks, each pairwise interaction corresponds to a single training example, and the interaction energy corresponds to the hinge loss on that example. Particle pairs that are separated with zero energy correspond to satisfied examples, whereas overlapping particle pairs with positive energy correspond to unsatisfied examples with positive hinge loss. The total energy over particle configuration space then corresponds to the loss function over neural network parameter space. Finally, particle density corresponds to the number ratio of examples P to network parameters N.

The jamming scenario exhibits an interesting phase transition between a low-density phase in which many particles are free to move (**Figure 2*d***) and a high-density jammed phase in which most pairwise interactions involve particle overlaps with positive energy (**Figure 2*e***). In the neural network analogy, the low-density phase corresponds to an overparameterized regime in which a small number of P examples can all be easily satisfied by a large number of N parameters, whereas the high-density jammed phase corresponds to an underparameterized regime in which a large fraction of P examples cannot be satisfied by a small number of N parameters. References 49, 50, and 52 explore this analogy between jamming and neural networks much more quantitatively, finding a host of intriguing phenomena, including the prevalence of many flat directions in the training error at the jamming transition and avalanche-like dynamics in which the set of unsatisfied examples exhibit rapid macroscopic rearrangements over training time. It would be interesting to understand to what extent the intuitions derived from this analogy extend to other loss functions beyond the hinge loss.

3.4. Explorations of Actual Neural Network Landscapes

In addition to the development and comparison of toy theoretical models for error landscapes, much work explored actual neural network landscapes. Recent work numerically explored the Hessian (53, 54) even for very large neural networks (55, 56). Interestingly, the Hessian near the bottom of the landscape after training exhibits a bulk spectrum that is heavy-tailed plus a set of outliers that are in one-to-one correspondence with the number of class labels in a classification task. Another interesting view of the error landscape was given in References 57 and 58, which suggested that the landscape contains rare but wide minima that are preferentially found by gradient descent, suggesting the possibility of new entropic algorithms that facilitate finding these minima (59). Much more controlled numerical work along these lines, concomitant with further theoretical developments, is required to attain a unified, high-resolution view of the shape of neural network error landscapes.

4. SIGNAL PROPAGATION AND INITIALIZATION IN DEEP NETWORKS

Before performing gradient descent on the loss landscape in Equation 3, one must choose an initial configuration of the parameters $\mathbf{w}$, corresponding to the entire set of weights and biases $\{\mathbf{W}^l, \mathbf{b}^l\}_{l=1}^{D}$ across all D layers in Equation 1. Often, these parameters are chosen randomly, with the weights $\mathbf{W}_{ij}^l$ chosen i.i.d. from a zero-mean Gaussian with variance σ_w^2/N_{l-1}, where N_{l-1} is the number of neurons in the presynaptic layer. The biases are chosen i.i.d. from a zero-mean

Gaussian with variance σ_b^2. The relative scaling of weights and biases ensures that both influence postsynaptic activity on an equal footing even for large N_{l-1}.

Natural questions then are, how should we choose the variances σ_w^2 and σ_b^2 to accelerate learning, and can non-Gaussian distributions of weights outperform Gaussian ones? In this section, we review how a theoretical understanding of signal propagation through such random deep networks has led to nontrivial initialization schemes that substantially outperform those often used previously in practice.

4.1. A Dynamical Phase Transition in Random Neural Networks

The theory of signal propagation through random networks simplifies in a large-width, mean-field limit in which N_l is large for all l. A connection between such wide networks and Gaussian processes was made in Reference 60 for single, hidden layer networks. In the large-width limit, we obtain the self-averaging property in which the empirical distribution of inputs $\mathbf{h}_i^l$ across neurons i into a layer l for fixed weights and biases equals the distribution of the input $\mathbf{h}_i^l$ for fixed i across randomly chosen weights and biases. Furthermore, for sufficiently regular nonlinearities ϕ, both distributions converge to a Gaussian as the width gets large (29, 61). In this limit, both the forward propagation of inputs and the back propagation of error signals exhibit a dynamical phase transition as a function of σ_w^2 and σ_b^2.

4.1.1. Forward propagation of inputs. To understand this phase transition from the perspective of forward-propagating inputs, consider a set of K input vectors $\{\mathbf{x}^{0,a}\}_{a=1}^K$, which will propagate forward to a set of input vectors $\{\mathbf{h}^{l,a}\}_{a=1}^K$ into layer l. We can describe the geometry of this point cloud by the matrix of inner products,

$$\mathbf{\Sigma}_{ab}^l = \frac{1}{N_l} \sum_{i=1}^{N_l} \mathbf{h}_i^{l,a} \mathbf{h}_i^{l,b}. \qquad 7.$$

In the large-width mean-field limit, one can track the geometry of this point cloud as it propagates through the layers by a set of deterministic recursion relations for computing $\mathbf{\Sigma}_{ab}^l$ in terms of $\mathbf{\Sigma}_{ab}^{l'}$ for $l' < l$, and these recursion relations only depend on σ_w^2, σ_b^2, and the nonlinearity ϕ (29, 31, 62). Such recursion relations have a large depth fixed point, which for fully connected networks with permutation symmetric input takes the form (29)

$$\lim_{l\to\infty} \frac{\mathbf{\Sigma}_{ab}^l}{\zeta(l)} = q^*[(1-c^*)\delta_{ab} + c^*]. \qquad 8.$$

Here, the function $\zeta(l)$ is an overall scaling function that accounts for unbounded growth of the inputs that can arise with unbounded nonlinearities or residual connections. In essence, at large depths any permutation-invariant point cloud converges to one in which the normalized length of all points is q^*, and the cosine angle between all pairs is c^*.

A small deviation $\delta\mathbf{\Sigma}^l = \mathbf{\Sigma}^l - \mathbf{\Sigma}^*$ about this fixed-point geometry obeys the linearized recursion relation $\delta\mathbf{\Sigma}^l \approx \boldsymbol{\chi}\delta\Sigma^{l-1}$. Thus, the dynamics of the point cloud exponentially converges (diverges) according to whether each eigenvalue λ of $\boldsymbol{\chi}$ is less (greater) than 1 in magnitude. For stable fixed points, in which all eigenvalues have magnitude less than 1, the relation $\lambda^l = e^{-l/\xi_\lambda}$ implicitly defines a depth scale ξ_λ over which the associated eigenmode of $\mathbf{\Sigma}^l$ converges to $\mathbf{\Sigma}^*$ (31). Fully connected networks possess two depth scales associated with the length q^* and the cosine angle c^*, with lengths converging more quickly than angles (29, 31). Convolutional networks possess different depth scales for different Fourier modes (63).

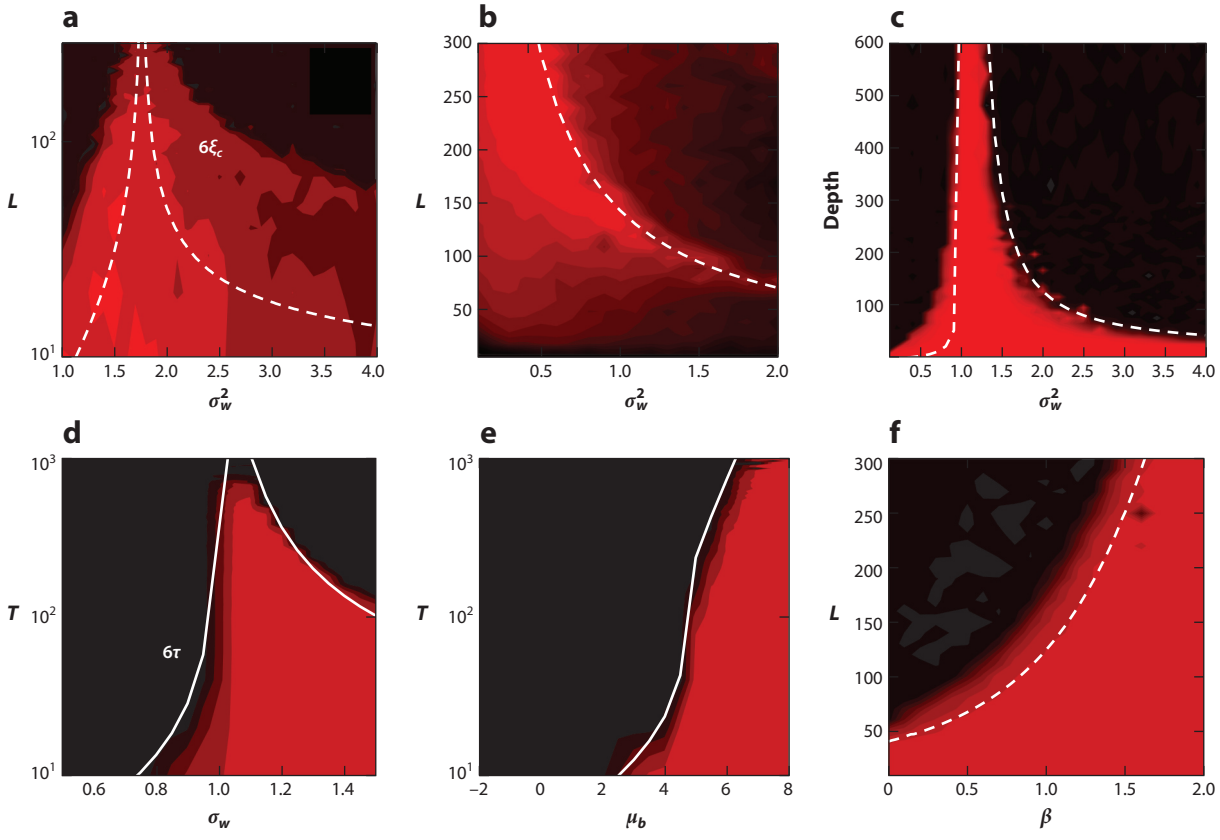

Figure 3

Signal propagation predicts trainability. Each panel shows training accuracy from perfect (*red*) to random chance (*black*) as the hyperparameters of a model are varied over a two-dimensional slice. White lines show mean-field predictions for quantities that determine trainability in each case. In general, we observe excellent agreement over a wide range of architectures. (*a*) Fully connected network compared with the depth scale for signal propagation. (*b*) A residual network compared with a curve of constant gradient norm. (*c*) Convolutional network with the depth scale for signal propagation. (*d–e*) Recurrent neural networks with the timescale for signal propagation. (*f*) Fully connected networks with batch normalization with the depth scale for gradient explosion. See Reference 31 for more details.

Fully connected networks with fixed points of the form in Equation 8 exhibit a phase transition as one increases σ_w^2 for fixed σ_b^2 for smooth bounded nonlinearities ϕ (see also **Figure 1*a***). For small σ_w^2, a fixed point with $c^* = 1$ is stable (implying all nearby points contract to a single point), whereas for large σ_w^2, this fixed point is unstable and another fixed point with $c^* < 1$ becomes stable (implying nearby points chaotically decorrelate to a nonzero angle, e.g., as in **Figure 1*b***). At the critical transition point, the depth scales ξ diverge, implying that forward propagation of signals retains a deep memory trace of the initial input geometry.

Intriguingly, this diverging depth scale for information propagation of input geometry coincides with the ability to train extremely deep, critical networks (31) (**Figure 3**). Furthermore, away from criticality, the depth scale of reliable forward information propagation controls the depth over which neural networks can be trained. This critical phase transition, diverging depth scale, and deep trainability at criticality have been observed not only in fully connected networks (31) but also in convolutional networks (63), autoencoders (64), and recurrent networks (65, 66). When

these same networks feature unbounded nonlinearities, such as ReLUs, they exhibit a phase transition (67) between a bounded phase, in which $\zeta(\ell) = 1$, and an unbounded phase, in which $\zeta(\ell) = \ell$. This transition comes from competition between the nonlinearity, which truncates some of the inputs to zero, and the weights, which can either expand or contract inputs depending on the value of σ_w^2. Overall, the theory of identifying critical initializations in which depth scales diverge has played a useful prescriptive role in the design of nonlinearities, initialization schemes, regularization schemes, and architectural choices to accelerate the training of extremely deep neural networks (31, 63–66, 68–71).

4.1.2. Back propagation of error signals. A key idea in training deep networks involves moving the weights $\mathbf{W}^l$ and, consequently, the activations $\mathbf{x}^l$ in each layer l so as to move the output $\mathbf{x}^D$ in the final layer in a desired direction. A fundamental linear operator determining how we should change $\mathbf{x}^l$ to move $\mathbf{x}^D$ is the susceptibility matrix or Jacobian $\frac{\partial \mathbf{x}^D}{\partial \mathbf{x}^l}$. This Jacobian is an important component of the back propagation of errors at the outputs to weights at layer l. For randomly initialized networks, this Jacobian is a random matrix, and its spectral properties are strongly correlated with the success or failure of training. Consider, for example, the full end-to-end input–output Jacobian,

$$\mathbf{J} = \frac{\partial \mathbf{x}^D}{\partial \mathbf{x}^0} = \prod_{l=1}^{S} \mathbf{D}^l \mathbf{W}^l. \qquad 9.$$

Here, $\mathbf{D}^l$ is a diagonal matrix with entries $D_{ij}^l = \phi'(h_i^l)\,\delta_{ij}$. This Jacobian determines how an error $\mathbf{e}$, or desired direction of motion in the output $\mathbf{x}^D$, back-propagates to a desired change in the input $\Delta \mathbf{x^0}^T = \mathbf{e}^T \mathbf{J}$. The growth incurred by back propagation is captured by $||\mathbf{e}^T \mathbf{J}||_2^2 / ||\mathbf{e}||_2^2$, which averages to $\mathrm{Tr}\, \mathbf{J}^T \mathbf{J}$ if $\mathbf{e}$ is a randomly chosen desired direction with i.i.d. components. In turn, in the infinite-width mean-field limit, this growth becomes self-averaging and does not fluctuate appreciably across networks. Therefore, we can define the growth rate χ via an average $\langle \cdot \rangle$ across the random network parameters $\{\mathbf{W}^l, \mathbf{b}^l\}_{l=1}^D$ in Equation 1:

$$\chi^D = \frac{1}{N_o} \left\langle \mathrm{Tr}\, \mathbf{J}^T \mathbf{J} \right\rangle, \qquad \text{where } \chi = \frac{1}{N_o} \left\langle \mathrm{Tr}\, (\mathbf{DW})^T (\mathbf{DW}) \right\rangle. \qquad 10.$$

Here, we have for simplicity chosen the same number of neurons $N_l = N_0$ for all layers l. χ is thus simply the mean-squared singular value of the Jacobian $\mathbf{DW}$ from one layer to the next. This local operator reflects an average multiplicative growth (shrinkage) of randomly chosen back-propagated errors $\mathbf{e}$ if $\chi < 1$ ($\chi > 1$), and this growth (or shrinkage) propagates exponentially with depth D.

The back propagation of errors is intimately related to the forward propagation of inputs, as shown in Reference 31 for fully connected networks. Thus, when the fixed point for forward propagation in Equation 8 with $c^* = 1$ is stable (i.e., the ordered regime in **Figure 1*a***), nearby input points come closer together as they propagate forward and back-propagated errors exponentially vanish. Conversely, when the $c^* < 1$ fixed point is stable (i.e., the chaotic regime in **Figure 1*a,b***), nearby input points diverge, and back-propagated errors exponentially explode. This picture can be generalized to other networks (62), including convolutional networks (63) and recurrent networks (65, 66). Again, initializing at a critical point, for example, at the boundary between the ordered and chaotic regimes, often leads to accelerated training and better final performance (31, 63–66, 68–71).

In addition to giving insights into initialization, a mean-field analysis of signal propagation and gradient back propagation has yielded insight into several other phenomena in deep learning.

These include the nature of adversarial examples (72), the eigenvalues of the Fisher information (73), the effect of weight quantization (74), and graph partitioning by graph neural networks (75).

4.2. Dynamical Isometry and Free Probability Theory

The previous section showed that the mean-squared singular value of the Jacobian **J** in Equation 9 grows as χ^D with χ defined in Equation 10. Therefore, critical initializations with $\chi = 1$ avoid exponential explosion or growth of a randomly chosen error signal **e**. However, this does not mean the worst-case largest growth or smallest shrinkage over all possible error signals **e** could not also grow or decay with depth. The largest growth and smallest shrinkage factors are controlled by the maximal and minimal singular values of **J**, respectively. Thus, one may conjecture that one could achieve even faster and better training through initializations that not only ensure the mean-squared singular value of **J** remains 1 but also that the entire singular value distribution of **J** remains tightly concentrated about 1. Such an initialization yields a property known as dynamical isometry, which was first introduced in Reference 76. Such an initialization ensures an isometric dynamic back propagation of errors in which the length of every error vector is approximately preserved, and angles between all pairs of error vectors are also preserved.

Achieving dynamic isometry in linear networks can be done simply by choosing weight matrices to be orthogonal rather than Gaussian, and it was shown both theoretically and empirically that orthogonal initialization actually achieves training times (measured in number of learning steps) that are independent of network depth (76). In contrast for Gaussian initializations even at criticality with $\chi = 1$, training times were linearly proportional to depth. Indeed, even with $\chi = 1$, products of Gaussian random matrices have a maximum singular value that grows linearly with depth, whereas products of orthogonal matrices have all singular values equal to 1, thereby achieving perfect dynamic isometry.

These results were generalized to nonlinear networks in References 77 and 78 using powerful tools from free probability theory (79, 80) to analytically compute the entire spectrum of products of random matrices underlying **J** as a function of the distribution of weights and the shape of the nonlinearity ϕ. This analytic theory matches numerical measurements of the empirical spectral distribution of **J** in nonlinear deep networks (**Figure 4*a***). Interestingly, this work revealed

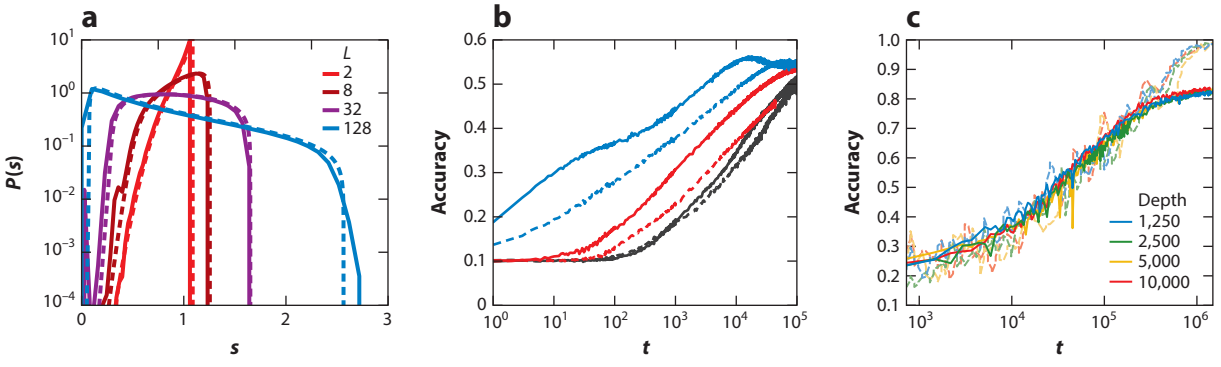

Figure 4

(*a*) Empirical singular value density of end-to-end Jacobians for erf networks of width 1,000 (*solid*) versus theory (*dashed*) for multiple depths. (*b*) Test accuracy dynamics on CIFAR-10 for networks of depth 200 and width 400. The different curves represent different nonlinearities and weight initializations, with the degree of dynamical isometry decreasing from blue to red to black. (*c*) Training accuracy (*dotted*) and test accuracy (*solid*) on CIFAR-10 for CNNs initialized with dynamical isometry for different depths. Training is possible up to depth 10,000. Abbreviations: CIFAR, Canadian Institute for Advanced Research; CNN, convolutional neural network.

that dynamical isometry can be achieved even in nonlinear networks with orthogonal weights and sigmoidal nonlinearities (or more generally nonlinearities that are locally linear near the origin with bounded derivatives elsewhere), but not with ReLUs, one of the most popular nonlinearities in deep learning. Furthermore, this work showed that no nonlinearity ϕ can achieve dynamical isometry if weights are Gaussian distributed (78). These theoretical results about the entire spectrum of $\mathbf{J}$ correlated with practical training results, which demonstrated sigmoidal networks with orthogonal initializations could actually outperform rectified linear networks (77) (**Figure 4*b***).

The conditions for dynamical isometry have been analyzed in a number of other architectures, including residual networks (81), RNNs (recurrent neural networks; 65), LSTMs/GRUs (long short-term memory networks/gated recurrent units; 66), and convolutional neural networks (CNNs; 63). In the case of CNNs, dynamical isometry enabled the trainability of extraordinarily deep networks consisting of 10,000 layers (**Figure 4*c***).

Thus, random matrix theory applied to deep networks has led to schemes for better training. In fact, random matrix theory has proven to be a powerful tool in deep learning in a variety of contexts, including in a study of the geometry of neural network loss surfaces (54, 82), in calculations of the spectrum of activation matrices (83) and of the Fisher information matrix (84), in investigations of learning dynamics (85–87), and in several other applications (88–90).

4.3. Beyond Mean Field: Finite Width and Path Integrals

The above theoretical results relied on two key simplifying assumptions: an infinite-width limit and i.i.d. weight and bias distributions. In such a mean-field limit, self-averaging holds, enabling the exact analytic computation of forward-propagating input geometry and Jacobian spectra for individual networks by averaging over network ensembles. To study the functional role of finite-width or trained networks (91, 92) we must go beyond mean field. Although research along these lines for deep learning is in its infancy, one can build upon a base of theoretical work studying finite-size effects in spin glasses (93), as well as path integral methods (94) for analyzing fluctuations in spin glasses (95), and stochastic (94) and deterministic (96–99) neural networks (see References 100 and 101 for reviews). Such path integral methods enable analytic computations of equations governing correlations and response functions (102) as well as a systematic treatment of fluctuations using Feynman diagrams and loop expansions (100, 101, 103–105).

Recently, such path integral methods have been employed to analyze trained feed-forward networks (91). However, this approach may have much further potential in shedding light on various aspects of deep networks. Indeed, just as this approach has been successful in elucidating the effects of finite-size corrections (99), correlations (106), and nonlinearities (107), in the case of recurrent networks, it may also yield similar insights beyond mean field in feed-forward networks (91, 108).

5. GENERALIZATION IN DEEP LEARNING

A critical question in deep learning involves understanding when and why deep networks generalize, so that only minimizing the training error $\mathcal{E}_{\text{Train}}(\mathbf{w}, \mathcal{D})$ in Equation 3 over the N parameters in $\mathbf{w}$ on a finite data set $\mathcal{D}$ of P examples still enables the network to accurately predict the correct answer to a new test input with low test error $\mathcal{E}_{\text{Test}}(\mathbf{w})$ in Equation 2. As we explain in this section, the capacity of deep networks to generalize remains a major theoretical puzzle (109), given the regime in which deep learning operates, where N can be orders of magnitude larger than P, with N sometimes even reaching billions of parameters (110).

5.1. Classical Theories of Generalization: Computer Science Versus Physics

Classical theories of generalization (111) in computer science frame any learning procedure as an algorithm $\mathcal{A}$ that maps any finite training set $\mathcal{D}$ to a particular function $f_{\mathcal{D}}$ in some space of functions $\mathcal{F}$: i.e., $\mathcal{A}(\mathcal{D}) = f_{\mathcal{D}} \in \mathcal{F}$. The data itself are often generated, perhaps stochastically, according to a ground truth function $f^* \in \mathcal{F}$, and generalization will be successful if in some sense $f_{\mathcal{D}}$ is close to f^*. Indeed, many theoretical computer science results on generalization prove bounds on the test error of the form

$$\mathcal{E}_{\text{Test}} \leq \mathcal{E}_{\text{Train}} + \frac{\mathcal{C}(\mathcal{F})}{P}, \qquad 11.$$

which hold with high probability over the choice of data set $\mathcal{D}$ for all ground truth functions $f^* \in \mathcal{F}$. Here, $\mathcal{C}(\mathcal{F})$ is some measure of the complexity of the function class $\mathcal{F}$. One prominent such measure is the Vapnik–Chervonenkis dimension (112), which for many neural networks grows with the number of parameters N. Alternatively $\frac{\mathcal{C}(\mathcal{F})}{P}$ in Equation 11 can be replaced by the Rademacher complexity $\mathcal{R}(\mathcal{F}, P)$, which measures how well a function chosen from $\mathcal{F}$ can correlate with, or memorize, P random noise patterns (113, 114). The intuition is if P is large and the function space $\mathcal{F}$ is not too complex, then $\mathcal{R}(\mathcal{F}, P)$ will be small, and generalization will be possible. An alternative framework for understanding generalization is the notion of algorithmic stability (115), which suggests that if the map $f_{\mathcal{D}} = \mathcal{A}(\mathcal{D})$ induced by the learning algorithm $\mathcal{A}$ is stable with respect to perturbations of the training data $\mathcal{D}$, then generalization will be successful. Finally, probably approximately correct (PAC) Bayes bounds suggest that if the distribution of weights does not change much during the training process, then generalization will be successful (116). The basic notions of function space complexity, algorithmic stability, and generalization are illustrated in **Figure 5*a***.

However, when many of these ideas are applied to large neural networks of size N trained on data sets of size P, with $N \gg P$ (a prominent regime in modern deep learning), they yield upper bounds on the test error that are exceedingly loose (117–123), an issue brought to the fore in Reference 109. However, it has been noted that a similar issue does indeed arise in kernel machines (124, 125), which can be thought of as two weight layer networks in which the first weight layer does not learn.

A parallel body of work on generalization in neural networks has played out in the physics literature. But in contrast to deriving upper bounds on test error, the focus in physics has been on asymptotically exact calculations of training and test errors in a thermodynamic limit in which both N and P become large but the measurement density $\alpha = P/N$ remains $O(1)$ (126–128) (see Reference 11 for an extensive review). In this framework, the data set $\mathcal{D}$ of P points $\{\mathbf{x}^{0,\mu}, \mathbf{y}^{\mu}\}_{\mu=1}^{P}$ is drawn from the random inputs and outputs of a teacher neural network with ground truth parameters $\mathbf{w}^*$. The training error $\mathcal{E}_{\text{Train}}(\mathbf{w}, \mathcal{D})$ is then thought of as an energy function over thermal degrees of freedom $\mathbf{w}$ of a student neural network, where the data $\mathcal{D}$ play the role of quenched disorder. The ground state of this statistical mechanical system on the student network parameters $\mathbf{w}$ is then compared with the ground truth teacher weights $\mathbf{w}^*$ to assess generalization. This approach has led to a rich body of work (11), but it can be difficult to carry out these calculations for complex modern neural networks as well as analytically demonstrate good generalization at tiny values of α for realistically structured data sets $\mathcal{D}$.

In the absence of adequate theory, many numerical explorations of generalization abound. One intriguing possibility is that good generalization is a kinetic, nonequilibrium dynamical property of SGD, which biases the learned parameters to highly flat regions of the training error (129, 130). Such flatness yields stability, which can yield generalization. Other approaches suggest weights do not accumulate much information about the training data, thereby yielding generalization (131)

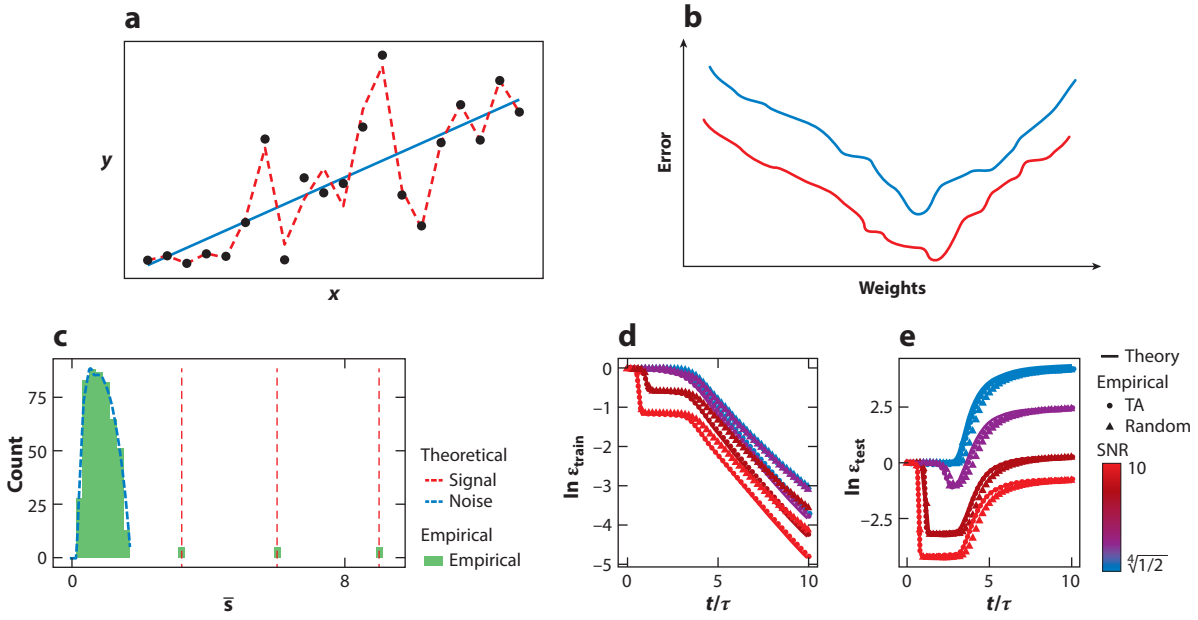

Figure 5

Generalization in deep learning. (*a*) Consider a finite data set $\mathcal{D}$ of P points (*black dots*) drawn from a ground truth linear function f^* plus noise. Minimizing the training error over a simple space $\mathcal{F}$ of linear functions yields a particular function (*blue line*) that is stable with respect to small perturbations of the training data and generalizes well to inputs from the training points. Conversely, minimizing over a more complex space $\mathcal{F}$ of piecewise linear functions yields a function (*red dashed line*) that fits the training set better but neither is stable nor generalizes well. (*b*) Because training corresponds to minimizing training error (*red curve*) and not the test error (*blue curve*), training error as a function of training time will monotonically decrease but test error may actually increase at late times, resulting in overfitting. (*c*) The singular value spectrum of the training data input–output covariance matrix $\mathbf{\Sigma}$ of a rank 3 linear teacher neural network. The singular modes associated with the three outlier singular values contain information about the teacher, whereas the remaining modes only contain information about noise in the finite training set (87). (*d*) Because learning corresponds to a singular mode detection wave that washes over the modes from right to left in panel *c*, the training error will drop as this wave passes a singular mode and will continue to drop as the wave penetrates the noise modes. (*e*) However, the test error will only drop when the detection wave passes over a signal mode; in contrast, the test error will rise as the wave later penetrates the noise modes (87). Abbreviations: SNR, signal-to-noise ratio; TA, teacher aligned.

(though see Reference 132). These works are related to the idea that learned neural networks might actually have a much smaller minimum description length (133, 134) than naive parameter counting might suggest. Of course, the full story is yet to be written.

5.2. Lessons from Linear Models

Given the complexity of the generalization problem, it is useful to glean lessons from simpler but nontrivial toy learning problems. An instructive learning problem is that of deep linear networks (76); despite the linearity of their input–output map, due to the composition of weights, the training and test error landscapes are highly nonconvex (**Figure 5*b***), yielding nonlinear learning dynamics that is, surprisingly, rich enough to even model many aspects of infant concept learning (9). Recent work derived closed-form solutions for the entire trajectory of both training and test errors (87) in the case of a low-rank teacher network and a full-rank student network. Thus, the student has many more parameters than the teacher yet it can still generalize well, as long as one stops the training early, which is a common practice in deep learning.

The analytic results reveal key conceptual insights into why good generalization is possible in such a scenario. First, the student learning dynamics builds up the singular value decomposition of the input–output correlation matrix $\mathbf{\Sigma} = \sum_\mu \mathbf{y}^\mu \mathbf{x}^{\mu^T}$ of the training data, learning the largest singular modes first and smaller singular modes later (76). Second, training data generated by a low-rank teacher yield a training data matrix $\mathbf{\Sigma}$ in which the modes of large singular value contain information about the teacher while the modes of smaller singular value contain information about irrelevant noise in the training data (**Figure 5*c***). Learning then corresponds to a singular-mode detection wave (87), which sweeps in from large to small singular values. As it crosses the large ones, the student learns about the teacher and both test and training errors drop, but as it crosses the small ones, the student only learns about irrelevant structure in the particular training set, and so training error drops but test error rises (**Figure 5*d***). Optimal early stopping then yields good generalization independent of the number of student network parameters N.

Extrapolating from this example, a natural conjecture is that real-world data sets $\mathcal{D}$ possess some underlying simple structure, and overparameterized deep neural networks tend to learn this simple structure first, thereby not making full use of all their parameters. Thus, good generalization may arise as a nonequilibrium kinetics conspiracy between structure in data and the propensity of deep network learning dynamics to preferentially learn this structure. It remains an intriguing theoretical question to identify what such simple structure may be in real-world data sets and whether this structure conspires with the intrinsic dynamics of deep learning to facilitate generalization, as it does in the deep linear case.

5.3. Lessons from the Infinite-Width Limit

Interesting work has noted that when the increase in the number of parameters N originates primarily from an increase in the network width, generalization actually tends to improve (67, 135–137). As we have seen in Section 4, signal propagation through randomly initialized neural networks has a well-defined limit as the width goes to infinity and the initial weight variance scales inversely with width. The observations in References 67 and 135–137 suggest that this infinite-width limit could also shed insight into the ingredients underlying successful generalization in large deep networks at the forefront of practice. Indeed, this infinite-width limit has well-defined learning properties, as we explain next.

5.3.1. Infinitely wide, deep neural networks are Gaussian processes. Any deep network of the form in Equation 1 can be viewed as a function $f\colon \mathbb{R}^{N_0} \to \mathbb{R}^{N_D}$. Randomness in the initial parameters $\mathbf{w}$ induces a prior distribution $\mathcal{P}(f)$ over the space $\mathcal{F}$ of such functions. In the infinite-width limit, this prior simplifies to become a Gaussian distribution, or process (60), over the space of all functions $f \in \mathcal{F}$ with a specific correlation kernel $\mathcal{K}$ (67, 138). We have already seen such smooth random Gaussian processes as a toy model for the error landscape $E(\mathbf{w})$ over weight space $\mathbf{w}$ in Section 3.1. In that setting, the function values E varied randomly over network parameters $\mathbf{w}$ with a correlation kernel between a pair of function values evaluated at two different parameters given by $\langle E(\mathbf{w}^\alpha)E(\mathbf{w}^\beta)\rangle = \mathcal{K}(\mathbf{w}^\alpha, \mathbf{w}^\beta)$. In this analogous setting, the activity of a single neuron $\mathbf{x}_i^D$ in layer D now varies randomly over the input space $\mathbf{x}^0$, with a correlation kernel between the activity evaluated at a pair of inputs $\mathbf{x}^{0,\alpha}$ and $\mathbf{x}^{0,\beta}$ given by $\langle \mathbf{x}_i^D(\mathbf{x}^{0,\alpha})\mathbf{x}_j^D(\mathbf{x}^{0,\beta})\rangle = \mathcal{K}_{ij}(\mathbf{x}^{0,\alpha}, \mathbf{x}^{0,\beta})$. The correlation kernel can be computed as a function of depth D through a sequence of deterministic recursion relations that depend only on the weight and bias variances σ_w^2 and σ_b^2 and the nonlinearity ϕ (67, 138). These recursion relations for the kernel are analogous to those for propagating input correlations, discussed in Section 4.1.1. Further work demonstrated that

infinitely wide randomly initialized convolutional networks are also characterized by a Gaussian prior over functions (137), with the number of convolutional channels playing the role of width.

5.3.2. Learning in the infinite-width limit. The process of learning in the infinite-width limit can be viewed as moving from a Gaussian prior $\mathcal{P}(f)$ over the function space $\mathcal{F}$ to a Bayesian posterior distribution $\mathcal{P}(f|\mathcal{D})$ conditioned on the training data $\mathcal{D}$. Indeed, the posterior $\mathcal{P}(f|\mathcal{D})$ is itself also a Gaussian process that can be computed explicitly (139, 140) through Gaussian integrals. Such a Bayesian posterior calculation in function space is reminiscent of calculations in field theory (e.g., see Reference 141).

Currently, the exact correspondence between learning using Bayesian inference on function space with a Gaussian process prior, corresponding to an infinitely wide deep network, and learning by gradient descent on a large but finite-width deep network of the types used in practice is not well understood. Empirical work for fully connected networks (67) found the two methods could perform similarly, hinting at a simple relationship between them. Indeed gradient-descent learning dynamics in infinitely wide deep neural networks was recently shown (142–144) to have a connection to kernel methods in which one fixes a large set of random features, or nonlinear functions over the input space $\mathbf{x}^0$, and only learns a linear combination of these fixed nonlinear functions. This picture holds for infinitely wide networks because it can be shown that gradient-descent dynamics does not appreciably move the weights (142, 143). However, it has been suggested (145) that the dynamics of References 142 and 143 is too simple to explain the empirical success of neural networks because such networks may change their weights much more during the learning process, hence learning many nonlinear basis functions adapted to the data, rather than linear combinations of fixed random nonlinear functions. Although early empirical results in Reference 143 identify, in some cases, finite-width networks with commonly used architectures whose dynamics are captured by the infinite-width theory, this issue is currently being investigated by the community.

The standard deviation of initial weights in the Gaussian process infinite-width limit (67, 142) scales inversely with the square root of width. Other scaling limits in infinitely wide networks have also been considered (146–148), where, for example, the final layer weights instead scale inversely with width. This limit gives rise to nontrivial interacting particle dynamics different from that in References 142 and 143. Further theoretical and empirical study is needed to identify which dynamical description(s) are most pertinent to neural networks as currently trained in practice and whether alternative scaling regimes would have advantages over current approaches. Overall, the study of generalization in deep learning is progressing rapidly, in part due to insights drawn from physical concepts like thermodynamic limits, interacting particle descriptions, and function space (field theoretic) formulations, and a more complete understanding may be on the horizon.

6. DEEP IMAGINATION THROUGH PROBABILISTIC MODELS

Although classic work in probabilistic unsupervised learning was often limited to fitting simple families of probability distributions $p(\mathbf{x}; \mathbf{w})$ to a data distribution $q(\mathbf{x})$ in Equation 4 by maximizing the log likelihood $l(\mathbf{w})$ of the data in Equation 5, recent advances in deep unsupervised learning have dramatically increased the complexity of distributions $p(\mathbf{x}; \mathbf{w})$ that can be fitted to data. These advances yielded striking applications in speech synthesis (17), representation learning and pretraining of models for other tasks (152), anomaly detection, inference of missing data, denoising (150), superresolution (153), compression (154), computer-aided design (155), and even nominally supervised tasks like classification and regression (156).

6.1. Energy-Based Probabilistic Models

Because they are most closely related to physics, we focus on energy-based probabilistic models in which $p(\mathbf{x}; \mathbf{w})$ is specified as a Boltzmann distribution (with the Boltzmann factor $kT = 1$):

$$p(\mathbf{x}; \mathbf{w}) = \frac{1}{Z_{\mathbf{w}}} e^{-E(\mathbf{x}; \mathbf{w})}. \quad 12.$$

The earliest energy-based probabilistic models in machine learning were in fact called Boltzmann machines (157), and map directly onto Ising spin models with a learned coupling structure $\mathbf{w}$. Extensive research has led to successively more sophisticated energy-based models (158–162), and they continue to be an active and competitive approach to generative and probabilistic modeling (150). Alternatives to energy-based models include autoregressive models and directed probabilistic graphical models.

6.2. Connections Between Learning, Information Theory, and Free Energy

Inserting the Boltzmann form in Equation 12 into the log-likelihood learning objective in Equation 5 yields

$$-l(\mathbf{w}) = \langle E(\mathbf{x}; \mathbf{w}) \rangle_q - F_{\mathbf{w}}, \quad 13.$$

where $\langle \cdot \rangle_q$ denotes an average with respect to the data distribution $q(\mathbf{x})$ and $F_{\mathbf{w}} = -\ln Z_{\mathbf{w}}$ is the Helmholtz free energy of the model distribution $p(\mathbf{x}; \mathbf{w})$. Thus, learning via maximizing the log likelihood corresponds to minimizing the energy of observed data while increasing the overall free energy of the model distribution.

Maximizing $l(\mathbf{w})$ is also equivalent to minimizing the Kullback–Leibler (KL) divergence,

$$D_{\mathrm{KL}}(q \| p) = \int d\mathbf{x}\, q(\mathbf{x}) \ln \frac{q(\mathbf{x})}{p(\mathbf{x}; \mathbf{w})} = G_{\mathbf{w}}(q) - F_{\mathbf{w}}. \quad 14.$$

Here, $D_{\mathrm{KL}}(q \| p)$ is a nonnegative information theoretic measure of the divergence between two distributions q and p that is zero if and only if $q = p$ (154). In the special case when p takes the Boltzmann form in Equation 12, the KL divergence becomes the difference between the Gibbs free energy of q, defined as $G_{\mathbf{w}}(q) = \langle E(\mathbf{x}; \mathbf{w}) \rangle_q - S(q)$ [where $S(q) = -\int d\mathbf{x}\, q(\mathbf{x}) \ln q(\mathbf{x})$ is the entropy of q] and the Helmholtz free energy $F_{\mathbf{w}}$ of p.

Learning corresponds to fixing the data distribution q and optimizing model parameters $\mathbf{w}$ in Equation 14. However, the decomposition in Equation 14 has another widespread application in both machine learning and statistical mechanics. Often, we are given a fixed complex Boltzmann distribution as in Equation 12 with coupling parameters $\mathbf{w}$, and we would like to approximate it with a simpler variational distribution $q(\mathbf{x})$. According to Equation 14, such an approximation can be derived by fixing $\mathbf{w}$ and equivalently minimizing with respect to q either the KL divergence $D_{\mathrm{KL}}(q \| p)$ or the Gibbs free energy $G_{\mathbf{w}}(q)$. Such an approach leads to both variational inference in machine learning and variational mean-field methods in physics.

6.3. Free Energy Computation as a Barrier to Learning

The previous subsection summarized tight relations between statistical concepts like log likelihoods and KL divergences, and physical concepts like energy, entropy, and free energies, thereby forming a bridge between machine learning and equilibrium statistical mechanics. In particular, for energy-based models, this bridge identifies the computation, approximation, and optimization of free-energy functions $F_{\mathbf{w}}$ as central to both fields. However, these are challenging problems for

both fields. Furthermore, in machine learning, even well-motivated approximations to $F_{\mathbf{w}}$ cease to be accurate over the course of training.

In the context of energy-based models, many approaches have been proposed to overcome the barrier of computing with free energies. These include exhaustive Monte Carlo, the contrastive divergence heuristic (163) and variants (164), score matching (165), pseudolikelihood (166), and minimum probability flow learning (MPF) (167, 168) (where MPF is itself grounded in nonequilibrium statistical mechanics). Sometimes, the requirement that a model be normalizable is relaxed and a probabilistic interpretation simply abandoned (169). Despite this progress, training expressive energy-based models on high-dimensional data sets remains an open challenge.

The difficulty of normalizing probability distributions over high-dimensional spaces (170) has led to interesting approaches for generative modeling of data that sidestep the computation of probabilities themselves. Such approaches include replacing explicit evaluation of probabilities with the judgement of a learned discriminator in generative adversarial networks (GANs)[1] (171), developing expressive classes of functions (related to Hamiltonian dynamics; 172) that can still be analytically normalized in the case of normalizing flows (173–175), factorizing the distribution into a product of one-dimensional conditional distributions in autoregressive models (176), and replacing posterior distributions with tractable variational approximations in variational autoencoders (177–180).

6.4. Nonequilibrium Statistical Mechanics

The bridge between machine learning and equilibrium statistical mechanics reviewed in Section 6.2 is just beginning to be extended to form links between machine learning and nonequilibrium statistical mechanics. In this section, we review two such links. However, this area is underexplored, and future research bridging nonequilibrium physics and machine learning seems likely to benefit both fields. Related promising directions include those that treat physical systems as information processing engines (181–184).

6.4.1. Jarzynski equality and annealed importance sampling. One of the most surprising analogs between machine learning and physics is that the Jarzynski equality (JE) is a special case of annealed importance sampling (AIS) from machine learning. Remarkably, the JE replaces the inequality in the second law of thermodynamics with an equality,

$$\exp\left(-\Delta F\right) = \langle \exp\left(-W\right) \rangle, \qquad 15.$$

where ΔF is the change in free energy between two macroscopic system states described by energy functions $E[x; \lambda(0)]$ and $E[x; \lambda(T)]$, $\lambda(t)$ describes time-dependent boundary conditions or control parameters interpolating between those states, W is the work done by moving along the trajectory $\lambda(t)$, and $\langle \cdot \rangle$ indicates an expectation over trajectories; we continue to assume the Boltzmann factor $kT = 1$.

AIS (185) and extensions (186, 187) are a generalization of importance sampling (IS), and allow an unbiased expectation to be computed over an intractable distribution by reweighting samples from a tractable distribution. In AIS, forward and reverse Markov chains are constructed bridging between the two distributions, allowing for a lower variance estimator than that provided by IS. If used to estimate the ratio of normalizers $\frac{Z^T}{Z^0}$ for two energy-based models $p\left[\mathbf{x}; \lambda(T)\right]$ and

[1]It has recently become popular to apply GANs to model physical systems. Extreme caution should be exercised when doing so. Their typical behavior is to generate high-quality but low-diversity samples, which neglect many aspects of the training distribution. This can easily lead to silently inaccurate conclusions.

$p[\mathbf{x}; \lambda(0)]$, AIS results in

$$\frac{Z^T}{Z^0} = \frac{Z^T}{Z^0} \int d\mathbf{x}^{0\ldots T} p_f\left(\mathbf{x}^{0\ldots T}\right) \frac{p_r\left(\mathbf{x}^{0\ldots T}\right)}{p_f\left(\mathbf{x}^{0\ldots T}\right)} = \left\langle \frac{e^{-E(\mathbf{x}^T;\lambda(T))}}{e^{-E(\mathbf{x}^0;\lambda(0))}} \prod_{t=1}^{T} \frac{p_r\left[\mathbf{x}^{t-1} \mid \mathbf{x}^t; \lambda(t-1)\right]}{p_f\left[\mathbf{x}^t \mid \mathbf{x}^{t-1}; \lambda(t-1)\right]} \right\rangle_{p_f}, \quad 16.$$

where $p_f\left(\mathbf{x}^{0\ldots T}\right) = p\left[\mathbf{x}^0; \lambda(0)\right] \prod_{t=1}^{T} p_f\left[\mathbf{x}^t \mid \mathbf{x}^{t-1}; \lambda(t-1)\right]$ and $p_r\left(\mathbf{x}^{0\ldots T}\right) = p\left[\mathbf{x}^T; \lambda(T)\right] \prod_{t=1}^{T} p_r\left[\mathbf{x}^{t-1} \mid \mathbf{x}^t; \lambda(t-1)\right]$ are the distributions over forward and reverse trajectories, respectively. In AIS, it is most common to choose the Markov transitions in the forward and reverse chains to satisfy a detailed balance condition such that $\frac{p_r\left[\mathbf{x}^{t-1}|\mathbf{x}^t;\lambda(t-1)\right]}{p_f\left[\mathbf{x}^t|\mathbf{x}^{t-1};\lambda(t-1)\right]} = \exp\left\{E\left[\mathbf{x}^t; \lambda(t-1)\right] - E\left[\mathbf{x}^{t-1}; \lambda(t-1)\right]\right\}$. If we further identify $\Delta W^t = E\left[\mathbf{x}^t; \lambda(t)\right] - E\left[\mathbf{x}^t; \lambda(t-1)\right]$ as the work done in time step t and $W = \sum_{t=1}^{T} \Delta W^t$ as the total work, and note that $\frac{Z^T}{Z^0} = \exp(-\Delta F)$, then we can see that the independently discovered equalities in Equations 15 and 16 become equivalent.

6.4.2. Nonequilibrium diffusion as a generative model. Ideas from nonequilibrium physics can be used not only to evaluate properties of a probabilistic model but also to define a probabilistic model. For example, in Reference 16 a parametric nonequilibrium process is trained to generate complex data distributions. The essential idea is to first slowly destroy structure in a complex data distribution $q\left(\mathbf{x}^0\right)$ by allowing individual data points to diffuse in data space. This diffusive process converts the complex, unknown data distribution $q\left(\mathbf{x}^0\right)$ into a simple, tractable distribution $p\left(\mathbf{x}^T\right)$ through a sequence of T iterative forward diffusion kernels $p_f\left(\mathbf{x}^{t+1} \mid \mathbf{x}^t\right)$ for $t = 0, \ldots, T-1$. For example, in the case of natural images, diffusion would correspond to each pixel intensity undergoing an independent unbiased random walk, which gradually turns any structured image into a white noise image.

One can then train a neural network to reverse the flow of time in this irreversible, entropy producing diffusive process. More precisely, each step of the time-reversing neural network is designed to move data points one step back in time by learning a reverse transition kernel $p_r\left(\mathbf{x}^t \mid \mathbf{x}^{t+1}\right)$. The composition of these kernels then yields a nonequilibrium generative model of the data. In this generative model, one just samples from the simple distribution $p\left(\mathbf{x}^T\right)$, and then repeatedly applies successive reverse transitions $p_r\left(\mathbf{x}^t \mid \mathbf{x}^{t+1}\right)$ to arrive at an approximation of the original data distribution $q\left(\mathbf{x}^0\right)$. For example, in the case of natural images, this process would correspond to sampling a white noise image and feeding it through a sequence of reverse transitions to arrive at a naturalistic image sample (16; **Figure 6*b***). Other examples of the ability

Figure 6

Physics-motivated probabilistic models have improved rapidly with the field of deep learning as a whole, but currently lag somewhat behind other probabilistic approaches, especially autoregressive models. All panels show samples from probabilistic models trained on a natural image data set. (*a–c*) Samples from physics-motivated probabilistic models; (*d*) samples from the current overall state-of-the-art (in terms of log likelihood) probabilistic model of images. Samples are from (*a*) an mcRBM energy-based model from 2010 (149), (*b*) the nonequilibrium diffusion model from 2015 described in Section 6.4.2 (16), (*c*) an energy-based model from 2019 (150), (*d*) the subscale pixel networks autoregressive model (151).

of neural networks to, nonintuitively, reverse the flow of time in irreversible stochastic processes can be found in Reference 16. Related work trains the parameters of both the forward and reverse processes (188) and breaks the strict correspondence to a diffusion process to allow sampling with fewer time steps (189).

7. SUMMARY

We hope this review conveys a sense of the progress surrounding the quest to obtain a theoretical understanding of the profound empirical success of deep learning. It is inevitable that our current theoretical understanding represents only the tip of the iceberg of a much more unified picture that will emerge over the ensuing years. However, it is exciting that even this small visible part has revealed a rich set of connections between the new field of deep learning and the comparatively ancient fields of statistical mechanics and condensed matter physics. Indeed, bread-and-butter topics in these fields, like random landscapes, phase transitions, chaos, spin glasses, jamming, random matrices, interacting particle systems, and nonequilibrium statistical mechanics, as well as more mathematical topics like free probability and Riemannian geometry, are beginning to shed light on intriguing phenomena in deep learning.

There are many opportunities for a judicious combination of controlled scientific experimentation on deep networks and the development of more realistic toy models of both training data and neural networks to deepen our existing understanding. Such a combination of experiment and theory has been a driving force for conceptual advances in physics, and we believe deep learning will provide more such research opportunities for physicists. Even more interestingly, this area of research may provide an opportunity for physicists to connect with computer scientists and neuroscientists as well as to develop a unified theory of how nonlinear distributed neural circuits, both artificial and biological alike, can compute, communicate, learn, and imagine (190).

DISCLOSURE STATEMENT

The authors are not aware of any affiliations, memberships, funding, or financial holdings that might be perceived as affecting the objectivity of this review.

ACKNOWLEDGMENTS

S.G. thanks the Burroughs Wellcome Fund and the Sloan, McKnight, James S. McDonnell, and Simons Foundations for support. J.K. thanks the Swartz Foundation.

LITERATURE CITED

1. LeCun Y, Bengio Y, Hinton G. 2015. *Nature* 521:436–44
2. Krizhevsky A, Sutskever I, Hinton GE. 2012. In *Advances in Neural Information Processing Systems 25 (NIPS 2012)*, ed. F Bereira, CJC Burges, L Bottou, KQ Weinberger, pp. 1097–105. Red Hook, NY: Curran Assoc.
3. Hannun A, Case C, Casper J, Catanzaro B, Diamos G, et al. 2014. arXiv:1412.5567
4. Devlin J, Chang MW, Lee K, Toutanova K. 2019. In *North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pp. 4171–86. Minneapolis, MN: Assoc. Comput. Linguist.
5. Silver D, Huang A, Maddison CJ, Guez A, Sifre L, et al. 2016. *Nature* 529:484–89
6. Yamins DLK, Hong H, Cadieu CF, Solomon EA, Seibert D, DiCarlo JJ. 2014. *PNAS* 111(23):8619–24
7. McIntosh L, Nayebi A, Maheswaranathan N, Ganguli S, Baccus S. 2016. See Reference 191, pp. 1369–77

8. Rogers TT, McClelland JL. 2004. *Semantic Cognition: A Parallel Distributed Processing Approach*. Cambridge, MA: MIT Press
9. Saxe AM, McClelland JL, Ganguli S. 2019. *PNAS* 116(23):11537–46
10. Piech C, Bassen J, Huang J, Ganguli S, Sahami M, et al. 2015. In *Advances in Neural Information Processing Systems 28 (NIPS 2015)*, ed. C Cortes, ND Lawrence, DD Lee, pp. 505–13. Red Hook, NY: Curran Assoc.
11. Engel A, den Broeck CV. 2001. *Statistical Mechanics of Learning*. Cambridge, UK: Cambridge Univ. Press
12. Mézard M, Montanari A. 2009. *Information, Physics, and Computation*. New York: Oxford Univ. Press
13. Advani M, Lahiri S, Ganguli S. 2013. *J. Stat. Mech. Theory Exp.* 2013:P03014
14. Mehta P, Bukov M, Wang CH, Day AGR, Richardson C, et al. 2019. *Phys. Rep.* 810:1–124
15. Carleo G, Cirac I, Cranmer K, Daudet L, Schuld M, et al. 2019. *Rev. Mod. Phys.* 91:045002
16. Sohl-Dickstein J, Weiss EA, Maheswaranathan N, Ganguli S. 2015. *Proc. Mach. Learn. Res.* 37:2256–65
17. van den Oord A, Dieleman S, Zen H, Simonyan K, Vinyals O, et al. 2016. arXiv:1609.03499
18. Nguyen HC, Zecchina R, Berg J. 2017. *Adv. Phys.* 66:197–261
19. Hornik K, Stinchcombe M, White H. 1989. *Neural Netw.* 2:359–66
20. Cybenko G. 1989. *Math. Control Signals Syst.* 2:303–14
21. Bengio Y, Courville A, Vincent P. 2013. *IEEE Trans. Pattern Anal. Mach. Intel.* 35:1798–828
22. DiCarlo JJ, Cox DD. 2007. *Trends Cogn. Sci.* 11:333–41
23. Montufar GF, Pascanu R, Cho K, Bengio Y. 2014. See Reference 192, pp. 2924–32
24. Delalleau O, Bengio Y. 2011. In *Advances in Neural Information Processing Systems 24 (NIPS 2011)*, ed. J Shawe-Taylor, RS Zemel, PL Bartlett, F Pereira, KQ Weinberger, pp. 666–74. Red Hook, NY: Curran Assoc.
25. Eldan R, Shamir O. 2015. *Proc. Mach. Learn. Res.* 49:907-940
26. Telgarsky M. 2015. *Proc. Mach. Learn. Res.* 49:1517–39
27. Martens J, Chattopadhya A, Pitassi T, Zemel R. 2013. In *Advances in Neural Information Processing Systems 26 (NIPS 2013)*, ed. CJC Burges, L Bottou, M Welling, Z Ghahramani, KQ Weinberger, pp. 2877–85. Red Hook, NY: Curran Assoc.
28. Bianchini M, Scarselli F. 2014. *IEEE Trans. Neural Netw. Learn. Syst.* 25:1553–65
29. Poole B, Lahiri S, Raghu M, Sohl-Dickstein J, Ganguli S. 2016. See Reference 191, pp. 3360–68
30. Sompolinsky H, Crisanti A, Sommers H. 1988. *Phys. Rev. Lett.* 61:259–62
31. Schoenholz SS, Gilmer J, Ganguli S, Sohl-Dickstein J. 2017. Paper presented at 5th International Conference on Learning Representations (ICLR 2017), Toulon, France. **https://openreview.net/forum?id=H1W1UN9gg**
32. Raghu M, Poole B, Kleinberg J, Ganguli S, Dickstein JS. 2017. *Proc. Mach. Learn. Res.* 70:2847–54
33. Mhaskar H, Liao Q, Poggio T. 2016. arXiv:1603.00988
34. Chung S, Lee DD, Sompolinsky H. 2018. *Phys. Rev. X* 8:031003
35. Boyd SP, Vandenberghe L. 2004. *Convex Optimization*. Cambridge, UK: Cambridge Univ. Press
36. Bray AJ, Dean DS. 2007. *Phys. Rev. Lett.* 98:150201
37. Fyodorov YV, Williams I. 2007. *J. Stat. Phys.* 129:1081–116
38. Dauphin YN, Pascanu R, Gulcehre C, Cho K, Ganguli S, Bengio Y. 2014. See Reference 192, pp. 2933–41
39. Baldi P, Hornik K. 1989. *Neural Netw.* 2:53–58
40. Kawaguchi K. 2016. See Reference 191, pp. 586–94
41. Choromanska A, Henaff M, Mathieu M, Arous GB, LeCun Y. 2015. *J. Mach. Learn. Res.* 38:192–204
42. Crisanti A, Sommers HJ. 1992. *Z. Phys. B Condens. Matter* 87:341–54
43. Crisanti A, Horner H, Sommers HJ. 1993. *Z. Phys. B Condens. Matter* 92:257–71
44. Auffinger A, Arous GB. 2013. *Ann. Probab.* 41:4214–47
45. Auffinger A, Arous GB, Černỳ J. 2013. *Commun. Pure Appl. Math.* 66:165–201
46. Baity-Jesi M, Sagun L, Geiger M, Spigler S, Arous GB, et al. 2018. *Proc. Mach. Learn. Res.* 80:314–23
47. Cugliandolo LF, Kurchan J. 1993. *Phys. Rev. Lett.* 71:173–76
48. Arous GB, Dembo A, Guionnet A. 2006. *Probab. Theory Relat. Fields* 136:619–60
49. Spigler S, Geiger M, d'Ascoli S, Sagun L, Biroli G, Wyart M. 2018. *J. Phys. A Math. Theor.* 52:474001

50. Geiger M, Spigler S, d'Ascoli S, Sagun L, Baity-Jesi M, et al. 2019. *Phys. Rev. E* 100:012115
51. O'Hern CS, Silbert LE, Liu AJ, Nagel SR. 2003. *Phys. Rev. E* 68:011306
52. Franz S, Parisi G. 2016. *J. Phys. A Math. Theor.* 49:145001
53. Sagun L, Bottou L, LeCun Y. 2016. Paper presented at 4th International Conference on Learning Representations (ICLR 2016), San Juan, Puerto Rico. arXiv:1611.07476
54. Sagun L, Evci U, Guney VU, Dauphin Y, Bottou L. 2018. Paper presented at 6th International Conference on Learning Representations (ICLR 2018), Vancouver, BC, Canada. arXiv:1706.04454
55. Papyan V. 2018. arXiv:1811.07062
56. Ghorbani B, Krishnan S, Xiao Y. 2019. In *Proceedings of the 36th International Conference on Machine Learning (ICML 2019), Long Beach, CA, June 9–15*, ed. K Chaudhuri, R Salakhutdinov, pp. 2232–41. Princeton, NJ: Int. Mach. Learn. Soc. arXiv:1901.10159
57. Baldassi C, Borgs C, Chayes JT, Ingrosso A, Lucibello C, et al. 2016. *PNAS* 113(48):E7655–62
58. Baldassi C, Ingrosso A, Lucibello C, Saglietti L, Zecchina R. 2015. *Phys. Rev. Lett.* 115(12):128101
59. Chaudhari P, Choromanska A, Soatto S, LeCun Y, Baldassi C, et al. 2017. Paper presented at 5th International Conference on Learning Representations (ICLR 2017), Toulon, France
60. Neal RM. 1996. *Bayesian Learning for Neural Networks*. New York: Springer Sci. Bus. Med.
61. Daniely A, Frostig R, Singer Y. 2016. See Reference 191, pp. 2253–61
62. Yang G. 2019. arXiv:1902.04760
63. Xiao L, Bahri Y, Sohl-Dickstein J, Schoenholz S, Pennington J. 2018. *Proc. Mach. Learn. Res.* 80:5393–402
64. Li P, Nguyen PM. 2019. Paper presented at 7th International Conference on Learning Representations (ICLR 2019), New Orleans, LA
65. Chen M, Pennington J, Schoenholz S. 2018. *Proc. Mach. Learn. Res.* 80:873–82
66. Gilboa D, Chang B, Chen M, Yang G, Schoenholz SS, et al. 2019. arXiv:1901.08987
67. Lee J, Bahri Y, Novak R, Schoenholz S, Pennington J, Sohl-Dickstein J. 2018. Paper presented at 6th International Conference on Learning Representations (ICLR 2018), Vancouver, BC, Canada
68. Yang G, Schoenholz S. 2017. See Reference 193, pp. 7103–14
69. Yang G, Pennington J, Rao V, Sohl-Dickstein J, Schoenholz SS. 2019. Paper presented at 7th International Conference on Learning Representations (ICLR 2019), New Orleans, LA
70. Pretorius A, van Biljon E, Kroon S, Kamper H. 2018. See Reference 194, pp. 5717–26
71. Hayou S, Doucet A, Rousseau J. 2018. arXiv:1805.08266
72. Cubuk ED, Zoph B, Schoenholz SS, Le QV. 2017. arXiv:1711.02846
73. Karakida R, Akaho S, Amari Si. 2018. arXiv:1806.01316
74. Blumenfeld Y, Gilboa D, Soudry D. 2019. arXiv:1906.00771
75. Kawamoto T, Tsubaki M, Obuchi T. 2018. See Reference 194, pp. 4361–71
76. Saxe A, McClelland J, Ganguli S. 2014. Paper presented at 2nd International Conference on Learning Representations (ICLR 2014), Banff, AB, Canada
77. Pennington J, Schoenholz S, Ganguli S. 2017. See Reference 193, pp. 4785–95
78. Pennington J, Schoenholz SS, Ganguli S. 2018. *Proc. Mach. Learn. Res.* 84:1924–32
79. Speicher R. 1994. *Math. Ann.* 298:611–28
80. Voiculescu DV, Dykema KJ, Nica A. 1992. *Free Random Variables*. Providence, RI: Am. Math. Soc.
81. Tarnowski W, Warchoł P, Jastrzebski S, Tabor J, Nowak MA. 2018. *Proc. Mach. Learn. Res.* 89:2221–30
82. Pennington J, Bahri Y. 2017. *Proc. Mach. Learn. Res.* 70:2798–806
83. Pennington J, Worah P. 2017. See Reference 193, pp. 2637–46
84. Pennington J, Worah P. 2018. See Reference 194, pp. 5410–19
85. Liao Z, Couillet R. 2018. *Proc. Mach. Learn. Res.* 80:3072–81
86. Advani MS, Saxe AM. 2017. arXiv:1710.03667
87. Lampinen AK, Ganguli S. 2018. Paper presented at 6th International Conference on Learning Representations (ICLR 2018), Vancouver, BC, Canada
88. Martin CH, Mahoney MW. 2018. arXiv:1810.01075
89. Louart C, Liao Z, Couillet R, et al. 2018. *Ann. Appl. Probab.* 28(2):1190–248
90. Liao Z, Couillet R. 2018. *Proc. Mach. Learn. Res.* 80:3072–81
91. Kadmon J, Sompolinsky H. 2016. See Reference 191, pp. 4781–89

92. Schoenholz SS, Pennington J, Sohl-Dickstein J. 2017. arXiv:1710.06570
93. Parisi G, Ritort F, Slanina F. 1999. *J. Phys. A Math. Gen.* 26:247
94. Martin PC, Siggia ED, Rose HA. 1973. *Phys. Rev. A* 8:423
95. Sommers HJ. 1987. *Phys. Rev. Lett.* 58:1268–71
96. De Dominicis C. 1978. *Phys. Rev. B Condens. Matter Mater. Phys.* 18:4913
97. Sompolinsky H, Crisanti A, Sommers HJ. 1988. *Phys. Rev. Lett.* 61:259–62
98. Kadmon J, Sompolinsky H. 2015. *Phys. Rev. X* 5(4):041030
99. Crisanti A, Sompolinksy H. 2018. *Phys. Rev. E* 98:062120
100. Hertz JA, Roudi Y, Sollich P. 2016. *J. Phys. A Math. Theor.* 50:033001
101. Schücker J, Goedeke S, Dahmen D, Helias M. 2016. arXiv:1605.06758
102. Janssen HK. 1976. *Z. Phys. B* 23:377–80
103. Chow CC, Buice MA. 2015. *J. Math. Neurosci.* 5:8
104. Buice MA, Cowan JD. 2007. *Phys. Rev. E* 75:051919
105. Buice MA, Chow CC. 2013. *J. Stat. Mech.* 2013:P03003
106. Martí D, Brunel N, Ostojic S. 2018. *Phys. Rev. E* 97:062314
107. Stapmanns J, Kühn T, Dahmen D, Luu T, Honerkamp C, Helias M. 2018. arXiv:1812.09345
108. Domany E, Meir R. 1991. In *Models of Neural Networks*, ed. E Domany, JL van Hammen, K Schulten, pp. 307–34. Berlin/Heidelberg: Springer-Verlag
109. Zhang C, Bengio S, Hardt M, Recht B, Vinyals O. 2017. Paper presented at 5th International Conference on Learning Representations (ICLR 2017), Toulon, France. arXiv:1611.03530
110. Shazeer N, Mirhoseini A, Maziarz K, Davis A, Le Q, et al. 2017. Paper presented at 5th International Conference on Learning Representations (ICLR 2017), Toulon, France. arXiv:1701.06538
111. Valiant LG. 1984. In *Proceedings of the 16th Annual ACM Symposium on Theory of Computing*, pp. 436–45. New York: Assoc. Comput. Mach.
112. Vapnik VN. 1998. *Statistical Learning Theory*. New York: John Wiley & Sons
113. Koltchinskii V, Panchenko D. 2000. In *High Dimensional Probability II*, ed. E Giné, DM Mason, JA Wellner, pp. 443–57. Boston: Birkhäuser
114. Bartlett PL, Mendelson S. 2002. *J. Mach. Learn. Res.* 3:463–82
115. Bousquet O, Elisseeff A. 2002. *J. Mach. Learn. Res.* 2:499–526
116. McAllester DA. 1999. In *Proceedings of the 12th Annual Conference on Learning Theory, (COLT 1999)*, ed. DA McAllester, pp. 164–70. New York: Assoc. Comput. Mach.
117. Bartlett PL, Mendelson S. 2002. *J. Mach. Learn. Res.* 3:463–82
118. Neyshabur B, Tomioka R, Srebro N. 2015. *Proc. Mach. Learn. Res.* 40:1376–401
119. Dziugaite GK, Roy DM. 2017. arXiv:1703.11008
120. Golowich N, Rakhlin A, Shamir O. 2018. *Proc. Mach. Learn. Res.* 75:297–99
121. Neyshabur B, Bhojanapalli S, Srebro N. 2018. Paper presented at 6th International Conference on Learning Representations (ICLR 2018), Vancouver, BC, Canada
122. Bartlett PL, Foster DJ, Telgarsky MJ. 2017. See Reference 193, pp. 6240–49
123. Arora S, Ge R, Neyshabur B, Zhang Y. 2018. *Proc. Mach. Learn. Res.* 80:254–63
124. Cortes C, Vapnik V. 1995. *Mach. Learn.* 20:273–97
125. Belkin M, Ma S, Mandal S. 2018. *Proc. Mach. Learn. Res.* 80:540–48
126. Gardner E. 1988. *J. Phys. A Math. Gen.* 21:257–70
127. Seung HS, Sompolinsky H, Tishby N. 1992. *Phys. Rev. A* 45:6056
128. Advani M, Ganguli S. 2016. *Phys. Rev. X* 6(3):031034
129. Hochreiter S, Schmidhuber J. 1997. *Neural Comput.* 9:1–42
130. Keskar NS, Mudigere D, Nocedal J, Smelyanskiy M, Tang PTP. 2017. Paper presented at 5th International Conference on Learning Representations (ICLR 2017), Toulon, France
131. Shwartz-Ziv R, Tishby N. 2017. arXiv:1703.00810
132. Saxe AM, Bansal Y, Dapello J, Advani M, Kolchinsky A, et al. 2018. Paper presented at 6th International Conference on Learning Representations (ICLR 2018), Vancouver, BC, Canada
133. Hinton G, Van Camp D. 1993. In *Proceedings of the 6th Annual Conference on Computational Learning Theory (COLT 1993)*, ed. L Pitt, pp. 5–13. New York: Assoc. Comput. Mach.

134. Hochreiter S, Schmidhuber J. 1994. In *Advances in Neural Information Processing Systems 31 (NIPS 1994)*, ed. S Bengio, H Wallach, H Larochelle, K Grauman, N Cesa-Bianchi, R Garnett, pp. 529–36. Red Hook, NY: Curran Assoc.
135. Neyshabur B, Tomioka R, Srebro N. 2015. Paper presented at 3rd International Conference on Learning Representations (ICLR 2015) Workshop Track, San Diego, CA, Abstr. #1412.6614
136. Novak R, Bahri Y, Abolafia DA, Pennington J, Sohl-Dickstein J. 2018. Paper presented at 6th International Conference on Learning Representations (ICLR 2018), Vancouver, BC, Canada
137. Novak R, Xiao L, Lee J, Bahri Y, Yang G, et al. 2019. Paper presented at 7th International Conference on Learning Representations (ICLR 2019), New Orleans, LA
138. de G. Matthews AG, Hron J, Rowland M, Turner RE, Ghahramani Z. 2018. Paper presented at 6th International Conference on Learning Representations (ICLR 2018), Vancouver, BC, Canada
139. Williams CK. 1997. In *Advances in Neural Information Processing Systems 10 (NIPS 1997)*, ed. MI Jordan, MJ Kearns, SA Solla, pp. 295–301. Red Hook, NY: Curran Assoc.
140. Rasmussen CE, Williams CKI. 2005. *Gaussian Processes for Machine Learning*. Cambridge, MA: MIT Press
141. Lemm J. 1999. arXiv:physics/9912005
142. Jacot A, Gabriel F, Hongler C. 2018. See Reference 194, pp. 8571–80
143. Lee J, Xiao L, Schoenholz SS, Bahri Y, Sohl-Dickstein J, Pennington J. 2019. In *Advances in Neural Information Processing Systems 32 (NIPS 2019)*, ed. S Bengio, H Wallach, H Larochelle, K Grauman, N Cesa-Bianchi, R Garnett, pp. 8570–81. Red Hook, NY: Curran Assoc.
144. Arora S, Du SS, Hu W, Li Z, Salakhutdinov R, Wang R. 2019. In *Advances in Neural Information Processing Systems 32 (NIPS 2019)*, ed. S Bengio, H Wallach, H Larochelle, K Grauman, N Cesa-Bianchi, R Garnett, pp. 8139–48. Red Hook, NY: Curran Assoc.
145. Chizat L, Bach F. 2018. See Reference 194, pp. 3036–46
146. Song M, Montanari A, Nguyen P. 2018. *PNAS* 115(33):E7665–71
147. Rotskoff GM, Vanden-Eijnden E. 2018. See Reference 194, pp. 7146–55
148. Sirignano J, Spiliopoulos K. 2019. *Stoch. Process. Appl.* In press
149. Ranzato M, Mnih V, Hinton GE. 2010. In *Advances in Neural Information Processing Systems 23 (NIPS 2010)*, ed. JD Lafferty, CKI Williams, J Shawe-Taylor, RS Zemel, A Culotta, pp. 2002–10. Red Hook, NY: Curran Assoc.
150. Du Y, Mordatch I. 2019. arXiv:1903.08689
151. Menick J, Kalchbrenner N. 2019. Paper presented at 7th International Conference on Learning Representations (ICLR 2019), New Orleans, LA
152. Radford A, Metz L, Chintala S. 2015. Paper presented at 3rd International Conference on Learning Representations (ICLR 2015), San Diego, CA
153. Zontak M, Irani M. 2011. In *Conference on Computer Vision and Pattern Recognition (CVPR 2011), Colorado Springs, CO, June 20–25*. Piscataway, NJ: IEEE. **https://doi.org/10.1109/CVPR.2011.5995401**
154. MacKay DJ. 2003. *Information Theory, Inference and Learning Algorithms*. Cambridge, UK: Cambridge Univ. Press
155. Zhu JY, Krähenbühl P, Shechtman E, Efros AA. 2016. In *European Conference on Computer Vision (ECCV 2016)*, ed. B Leibe, J Matas, N Sebe, M Welling, pp. 597–613. Cham: Springer
156. Murphy KP. 2012. *Machine Learning: A Probabilistic Perspective*. Cambridge, MA: MIT Press
157. Ackley DH, Hinton GE, Sejnowski TJ. 1985. *Cogn. Sci.* 9:147–69
158. Freund Y, Haussler D. 1992. In *Advances in Neural Information Processing Systems 5 (NIPS 1992)*, ed. SJ Hanson, JD Cowan, CL Giles, pp. 912–19. Red Hook, NY: Curran Assoc.
159. Hinton GE, Osindero S, Teh YW. 2006. *Neural Comput.* 18:1527–54
160. Salakhutdinov R, Hinton G. 2009. *J. Mach. Learn. Res.* 5:448–55
161. Ngiam J, Chen Z, Koh PW, Ng AY. 2011. In *Proceedings of the 28th International Conference on Learning Representations (ICLR 2011)*, ed. L Getoor, T Scheffer, pp. 1105–12. Madison, WI: Omnipress
162. Zhao J, Mathieu M, LeCun Y. 2017. Paper presented at 5th International Conference on Learning Representations (ICLR 2017), Toulon, France
163. Hinton GE. 2002. *Neural Comput.* 14:1771–800

164. Tieleman T, Hinton G. 2009. In *Proceedings of the 26th International Conference on Machine Learning (ICML 2009), Montreal, Quebec, Canada, June 14–18*, ed. A Danyluk, L Bottou, M Littman, pp. 1033–40. New York: Assoc. Comput. Mach.
165. Hyvärinen A. 2005. *J. Mach. Learn. Res.* 6:695–709
166. Besag J. 1975. *J. R. Stat. Soc. Ser. D (Statistician)* 24:179–95
167. Sohl-Dickstein J, Battaglino P, DeWeese MR. 2011. In *Proceedings of the 28th International Conference on Machine Learning (ICML 2011), Bellevue, Washington, June 28–July 2*, ed. L Getoor, T Scheffer, pp. 905–12. Madison, WI: Omnipress
168. Sohl-Dickstein J, Battaglino P, DeWeese MR. 2011. *Phys. Rev. Lett.* 107:220601
169. LeCun Y, Chopra S, Hadsell R, Ranzato M, Huang FJ. 2006. In *Predicting Structured Data*, ed. G Bakır, T Hofmann, B Schölkopf, A Smola, B Taskar, pp. 191–246. Cambridge, MA: MIT Press
170. Jordan MI. 2003. *An Introduction to Probabilistic Graphical Models*. Chapters available at **https://people.eecs.berkeley.edu/~jordan/prelims**
171. Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, et al. 2014. See Reference 192, pp. 2672–80
172. Levy D, Hoffman MD, Sohl-Dickstein J. 2017. Paper presented at 5th International Conference on Learning Representations (ICLR 2017), Toulon, France
173. Dinh L, Krueger D, Bengio Y. 2014. arXiv:1410.8516
174. Dinh L, Sohl-Dickstein J, Bengio S. 2016. Paper presented at 4th International Conference on Learning Representations (ICLR 2016), San Juan, Puerto Rico
175. Rezende DJ, Mohamed S. 2015. Paper presented at 3rd International Conference on Learning Representations (ICLR 2015), San Diego, CA
176. van den Oord A, Kalchbrenner N, Kavukcuoglu K. 2016. *Proc. Mach. Learn. Res.* 48:1747–56
177. Kingma DP, Welling M. 2014. Paper presented at the 2nd International Conference on Learning Representations (ICLR 2014), Banff, AB, Canada
178. Gregor K, Danihelka I, Mnih A, Blundell C, Wierstra D. 2014. *Proc. Mach. Learn. Res.* 32(2):1242–50
179. Rezende DJ, Mohamed S, Wierstra D. 2014. *Proc. Mach. Learn. Res.* 32(2):1278–86
180. Ozair S, Bengio Y. 2014. arXiv:1410.0630
181. Crutchfield JP, Mitchell M. 1995. *PNAS* 92:10742–46
182. Still S, Sivak DA, Bell AJ, Crooks GE. 2012. *Phys. Rev. Lett.* 109:120604
183. Parrondo JM, Horowitz JM, Sagawa T. 2015. *Nat. Phys.* 11:131–39
184. Lahiri S, Sohl-Dickstein J, Ganguli S. 2016. arXiv:1603.07758
185. Neal RM. 2001. *Stat. Comput.* 11:125–39
186. Neal RM. 2005. arXiv:math/0511216
187. Sohl-Dickstein J, Culpepper BJ. 2012. arXiv:1205.1925
188. Goyal A, Ke NR, Ganguli S, Bengio Y. 2017. See Reference 193, pp. 4392–402
189. Bordes F, Honari S, Vincent P. 2017. Paper presented at 5th International Conference on Learning Representations (ICLR 2017), Toulon, France
190. Gao P, Ganguli S. 2015. *Curr. Opin. Neurobiol.* 32:148–55
191. Lee DD, Sugiyama M, Luxburg UV, Guyon I, Garnett R, eds. 2016. *Advances in Neural Information Processing Systems 29 (NIPS 2016)*. Red Hook, NY: Curran Assoc.
192. Ghahramani Z, Welling M, Cortes C, eds. 2014. *Advances in Neural Information Processing Systems 27 (NIPS 2014)*. Red Hook, NY: Curran Assoc.
193. Guyon I, Luxburg UV, Bengio S, Wallach H, Fergus R, et al., eds. 2017. *Advances in Neural Information Processing Systems 30 (NIPS 2017)*. Red Hook, NY: Curran Assoc.
194. Bengio S, Wallach H, Larochelle H, Grauman K, Cesa-Bianchi N, Garnett R, eds. *Advances in Neural Information Processing Systems 31 (NIPS 2018)*. Red Hook, NY: Curran Assoc.