Machine learning meets physics: A two-way street

Herbert Levine[a] and Yuhai Tu[b,1]

This article introduces a special issue on the interaction between the rapidly expanding field of machine learning and ongoing research in physics. The first half of the papers in this issue deals with the question, what can machine learning do for physics? The second part asks the reverse, what can physics do for machine learning? As we will see, both of these directions are being vigorously pursued.

Physics is, of course, a very broad discipline, and almost every part of it has been exploring the possible use of machine learning (ML). We obviously cannot cover all of these developments systematically. Instead, we will present various examples, and try to propose some tentative general insights. Given the tremendous buzz of activity, we are sure that our perspective will need to be constantly revised in the light of accumulating experience. Nevertheless, we proceed.

1. ML and the Protein Folding Problem

The paradigmatic example of machine learning solving an important physics problem is the performance of AlphaFold (1) and its successors in determining protein structure from sequence. This is a problem that has been studied in the biophysics discipline for many years (2–4), with the community actively participating in a biannual contest known as CASP (the Critical Assessment of Structural Prediction) in which methods are evaluated versus known but as yet unrevealed data. A milestone was reached in 2018 when AlphaFold placed first in the overall rankings, and this was repeated in 2020. By 2022 in CASP15, most of the entrants had adopted some form of AlphaFold-based ideas for use in their methods. The method has become so pervasive that the word has begun to be used as a verb, as in "Can we AlphaFold our way out of the next pandemic?" (5). The paper in this special issue by Park et al. (6) provides a useful guide as to how to effectively use AlphaFold2 on modern computing systems.

It is quite interesting to consider the lessons one can garner from the history of the protein folding application. Both of us were at UCSD in the late 1980s when Terry Sejnowski presented a seminar on his paper using neural networks to study the secondary structure of globular proteins (7). The performance of his algorithm was rather mediocre, and many of us left that talk wondering why would anyone forgo traditional methods in biochemistry in favor of what we now call ML. So, what happened in the 30 y from then to 2018? There appear to have been four factors at work. First, the computational power able to be brought to bear on this problem has increased at an almost unimaginable rate. For example, the Cray 2 supercomputer circa 1985 reached 1.9 gigaflops as the world's fastest in that era; it is now roughly the equivalent of the iPhone 4. The current supercomputer leader reaches 10^9 gigaflops. What was computationally impossible then is totally trivial now. One could provide similar data regarding memory capacity. It is fair to say that progress would have been greatly hampered had the computing power not been available to the relevant researchers.

Hardware improvements are necessary but not sufficient. The second factor is the invention of a wide variety of ML techniques for learning predictive models from available data. In the 1980s, the field of neural networks was still in the throes of despair, brought about by the lack of understanding of the severe limitations of the famous "no-go" results of Minsky and Papert in their work on perceptrons (8). The idea that building "deep networks" with hidden layers would create a new path forward was just beginning to be realized, initially with constructs such as the Boltzmann machine (9) and then the formulation of the backpropagation training algorithm (10). Nowadays, ideas such as the transformer architecture (11), autoencoders (12), and adversarial networks (13) have revolutionized how one understands the ML process. For our example of AlphaFold, the transformer idea appears to be absolutely essential. A brief guide to how transformers fit in with more general machine learning concepts dating all the way back to the Hopfield associative memory model (14) is discussed in the paper by Martin et al. (15).

The next factor is data availability for training. The protein databank (PDB) was established in 1971 to store information regarding protein structure (16). Again, there has been an absolute explosion in the amount of structural data available to all researchers. The number of structures has doubled approximately every 6 to 8 y; at the beginning of 2024, the PDB archive surpassed 200K structures as compared to roughly 1000 in 1990; see Fig. 1. But this is not the only important source of data. As described in the article by Martin et al. (15), much of the progress in protein folding arose from the recognition that comparing sequences of the same protein in different organisms could enable one to obtain important information regarding the contact map. The contact map is a matrix representation of the chance that residues located some distance apart along the backbone were likely in the folded structure to be nearby

Author affiliations: [a]Center for Theoretical Biological Physics, Northeastern University, Boston, MA 02115; and [b]IBM T. J. Watson Research Center, Yorktown Heights, New York, NY 10598

Author contributions: H.L. and Y.T. designed research; performed research; analyzed data; and wrote the paper.

The authors declare no competing interest.

[1]To whom correspondence may be addressed. Email: yuhai@us.ibm.com.

Published June 24, 2024.

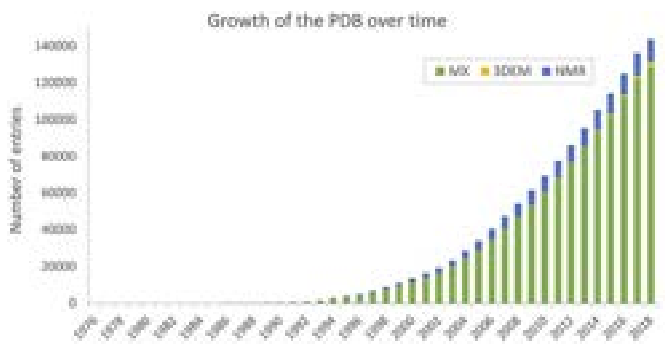

Fig. 1. A 2019 report on the growth of the PDB Core Archive. Total height of each bar indicates aggregate released structures, made up of subbars labeled by the experimental technique (MX (Crystallography)–green, 3DEM–yellow, NMR–blue) Taken from ref. 16.

in three-dimensional space. The essential idea here, arising from work in algorithms such as direct coupling analysis (DCA) (17, 18), is that a pair of residues in contact will have to coevolve to maintain that contact, as one goes from species to species. Thus, the observation of correlated evolution can help identify these contacts. The last decade has seen a tsunami of growth in comparative genomics data due to the amazing technological advances in the sequencing field.

Finally, we reach perhaps the most interesting factor-related question, one with important consequences for the ML research program moving forward. The question is, how essential was three decades of extensive theoretical research into protein folding using more traditional techniques? Put more simply, in an alternate universe where no one had paid attention to protein folding computations until the era of AlphaFold, how far behind our current state of affairs would we be? It is of course impossible to know the answer, but our sense is that the theory did play an important role in enabling current progress. We have already mentioned that the idea of using comparative genomics data arose in the theory community. It is also important to note the formulation of the idea of encoding structural data into standard biophysical models (19), taking advantage both of physical insight and measured information. Also, we should not confuse advancing the engineering use of protein folding in predicting structures from sequence, with obtaining a better understanding of protein folding through concepts such as minimal frustration (20) and the folding funnel (21). These latter concepts are globally important ideas that have found use in other contexts, both at the molecular (22, 23) and cellular (24) levels. And, it is sometimes nice to have "human interpretable" methods, even if it is they are unnecessary from a purely practical perspective.

What are the current challenges for this line of research? One can point to systems where there is not one unique structure, but instead, the folding problem translates to finding an ensemble of structures and the concomitant dynamics of transitions among them. These systems include intrinsically disordered proteins (25) as well as folding of the genome (26, 27). Another direction concerns the study of interacting biomolecules, where the pure ML approach of Alpha-Multimer has not yet proven reliable enough for many applications. The paper by Lupo et al. (28) attempts to address this problem by applying a language model to better align relevant interacting sequences at the protein–protein interface. One should also note the problem of antigen recognition by T cell receptors as a critical part of the adaptive immune system. Here, recent works (29, 30) approach this problem by the use of language models, competing with mixed approaches that incorporate structural data (31) and are therefore limited by the lack of extensive such data.

2. The Spreading of ML

Biological physics is a natural avenue for exploring the use of ML. Unlike many other areas of physics, most experimental systems related to the living world are exceedingly complex and hence the ability to form first-principles models rather limited. To pick an example from a larger scale than molecular, there are no first-principles models of collective cell motion (32) that can possibly do justice to the full complexity of the cell machinery employed for this behavior. There is no Navier–Stokes equation rushing to the rescue and one can therefore naturally wonder whether hand-crafted models (33, 34) could perhaps be usefully replaced by purely data-driven ones. This question is being actively investigated for a number of experimental cell motility systems (35, 36) and of course is under intense study in many biomedical contexts, see e.g. work on digital pathology (37). It is worth noting that one could try to derive better hand-crafted models themselves by ML (see e.g. ref. 38); it is not clear why this is better than just directly using the predictions from a learned neural network.

Given the above, it is perhaps more surprising that ML methodology is infiltrating into the study of physical systems which nominally do have reliable computational frameworks. An excellent summary of these varied systems is given in the paper by Yu and Wang (39). One idea posits that ML can speed up computations even when a first-principles model is available. One such claim is presented by Kochkov et al. (40), focusing explicitly on the aforementioned Navier–Stokes equation for fluid dynamics. Perhaps a more convincing case can be made for cases when the physics is knowable in principle but may be too complex to implement; one can think of cloud models in climate simulators as perhaps one such example. From a general perspective, it seems like there is much progress to be made in figuring out the optimum way of combining the interpretability of traditional modeling with the generalizability of ML methods.

If one is interested in a model of a very specific physical system, one can often afford to do the necessary large-scale computations to obtain meaningful results; and, this gets easier as computing power continues to grow exponentially. However, as emphasized in the paper in this issue by King et al. (41) in the context of material assembly, this becomes much more difficult when the task is to design something new. This challenge requires an iterative process of picking interactions at a microscopic scale which ultimately gives rise to some functional behavior at a much large scale. This iterative process often involves some type of gradient-descent using some functionality measure, but the "forward" problem must be computed many times as part of the convergence process. As discussed in this paper, this problem can be greatly assisted by machine learning ideas, including the concept of automatic differentiation (42) which enables the "backpropagation" of large-scale errors to the necessary changes in microscopic degrees. Of course, this idea was at the heart of the training algorithm for hidden layers in neural network models, but now, this idea can be applied automatically to any large-scale computation.

When one thinks about ML and its applications to physics, it is unlikely that string theory is something that comes immediately to mind. Yet, the string theory community is actively exploring whether ML methods can be useful (43). String theory, of course, is an attempt at formulating the "theory of everything," explaining all the elementary particles and their interactions in terms of "strings" (quantum objects extended in one dimension) living in 11 dimensions. ML is being used to find ways of compacting this 11-dimensional space into the 4-dimensional world that we experience and searching for a reasonable compactification is a very hard computational problem that can be ameliorated by ML ideas. Who knew?

There is one last direction being investigated at the ML-Physics interface. Some groups are attempting to use ML methods to automatically discover new equations from data; imagine taking planetary data and trying to learn Newton's laws of motion together with the inverse square law of gravitational force. This idea is briefly sketched in the paper by Yu (39), with relevant references. One can think of this endeavor as attempting to ultimately replace theoretical physicists with AI versions thereof. Until we see a machine that can look at astrophysical data and figure out that the correct framework for understanding is Riemannian geometry in 4-dimensional space-time, we are not worried about our jobs.

3. What Can Physics Do for ML?

Of course, the impact of ML goes well beyond its use in advancing physical science. Deep learning neural network (DLNN) models (44, 45) have enjoyed a long string of rapid and tremendous successes in image recognition (46), machine translation (47), games (48), and as we have already discussed, even solving long-standing grand challenge scientific problems such as protein folding (1). For better or worse, the most recent generative models such as ChatGPT, are fundamentally changing the social, economical, and political landscapes of our time.

However, one of the side-effects of the incredible success recently demonstrated by DLNNs has been to lose sight of its theoretical motivations and underpinning in favor of rapid narrowly focused application-driven developments. This is slowly resulting in increasingly suboptimal practices, including massive wastes in compute cycles and time to tune the large numbers of hyperparameters admitted by unprincipled optimization and regularization procedure, inefficient utilization of high-precision encoded parameters, inefficient utilization of costly labeled data, lack of reproducibility of end results, and the possibility of misuse of this powerful technology. The development of a principled theoretical foundation of overparameterized connectionist machine learning models like deep learning neural networks would help avoid such problems, thereby streamlining their optimization and allowing for robust models trained on less data. At the same time, the predictions provided by a prescriptive theory could guide the development of improved future architectures and training paradigms.

Artificial neural network (ANN) models originated from the marriage of two natural science disciplines—statistical physics and neuroscience. At their core, ANNs describe the emergent (collective) behaviors of a group of highly abstracted "neurons" interacting with each other in an adaptive way in a network that bears a certain resemblance to the real neural network in the brain. The model dynamics allow the ANNs to associate and learn. Historically, both statistical physics and neuroscience played a seminal role in the inception and early developments of ANNs. The linear–nonlinear artificial neurons and the synaptic weights between neurons as first introduced by McCulloch and Pitts (49) in 1943 for modeling a biological neural network are still the fundamental building blocks of the modern-day deep learning neural networks. Statistical physics also played an important role in the initial development of artificial neural networks and their theoretical understanding in the late 80s and 90s, motivating critical developments such as the Hopfield model (14), the Boltzmann machine (9), and applications of spin-glass theory to neural networks (50).

What is different at the current time? At the elemental level, not much, the McCulloch–Pitts neurons are still the building blocks of all deep learning algorithms, and the linear summation and nonlinear activation is still the basic computing process at the single neuron level. However, the scale is vastly different. As already discussed above in the protein folding context, we now have a huge amount of data with which to train large ANN models; in turn, these models can absorb the information in these large datasets by using a huge number of parameters. The architecture of these large models is much more sophisticated than that of the original perceptron model by Rosenblatt (51), e.g., the transformer architecture is critical for modern large language models (LLM). And of course, the performance of these large ANNs far exceeds our expectations based on looking at individual neurons.

This reminds us of the famous quote by P. W. Anderson: "More is different" (52), where he advocated the idea that the whole system is not only more than the sum of its parts but emergent (different) behaviors can arise due to interactions of the individual parts in the system. Just as the Anderson quote has encouraged generations of physicists to study emergent behaviors of complex many-body systems, we would like to use it as a rallying cry for physicists to study this fascinating emergent behavior called learning in the (sometimes) large but always well-structured artificial neural networks. These studies would have to answer general questions on how learning arises from interactions of neurons in DLNNs, what the networks learn, and whether they can generalize their learned knowledge.

Indeed, we believe that the next breakthroughs in deep learning may come from developing a solid theoretical foundation, based on concepts and methods from statistical physics. This will occur in conjunction with the introduction of ever-more advanced DLNN algorithms, algorithms that will accelerate the rate of scientific discovery in the physical and biological world. These two interconnected emerging topics of research, fundamental theory and sophisticated applications, will greatly advance both science and AI technology. In the following, we present a general framework for describing the machine learning process before delving into a few promising directions where progress may be made. Our discussion contains brief sketches of papers published in this special issue that are relevant to these directions.

3.1. The Central Dogma of Machine Learning. In the book "How learning works?" by Ambrose et al. (53), learning is defined as "a process that leads to change, which occurs as a result of experience and increases the potential for improved performance and future learning". The book was written in the context of human (student) learning, but this succinct definition of learning can be used to describe machine learning as well. In Fig. 2, we illustrate the key components and the workflow of machine learning such as neural-network-based deep learning, which we call the "Central Dogma" of Machine Learning. The goal of the machine learning process is to learn a model that captures the intrinsic properties of the external world represented by the observed data. The model has certain structure, i.e., functional form, and is parameterized by its parameters (weights in the context of neural network models). Following the definition given by Ambrose et al, during the training phase of the learning process, the parameters in the model change, which occurs as a result of training over experience or equivalently training data as called in machine learning. Once trained, the quality of learning can be evaluated by the performance of the trained model on unseen test data and whether the trained model forms a good basis (starting point) for future learning.

The workflow of machine learning as illustrated in Fig. 2 immediately suggests two important problems in machine learning. The first problem focuses on learning dynamics. More specifically, how do the parameters of the model change given the training data? The usual learning process is carried out by minimizing a loss function that characterizes how well the model fits the training data. Starting with an initial set of parameter values, the parameters are updated iteratively through the high-dimensional parameter space, guided by the loss function until they reach a minimum

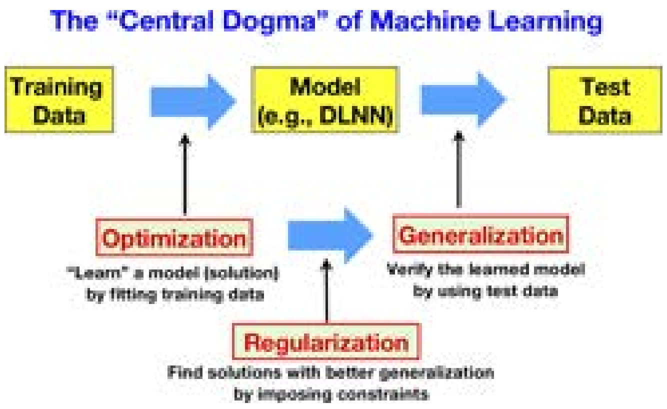
The "Central Dogma" of Machine Learning
Training Data
Model (e.g., DLNN)
Test Data
Optimization
"Learn" a model (solution) by fitting training data
Generalization
Verify the learned model by using test data
Regularization
Find solutions with better generalization by imposing constraints

Fig. 2. Illustration of the main steps and workflow in machine learning. Highlighted in red are three directions that may be addressed by physics-based approaches.

thereof. The model with its parameters given at such a minimum is a solution to the problem. The optimization process, i.e., the sequence of parameter updates, can be considered as dynamics of learning with an update step taken as time. The second problem deals with generalization. Typically, DLNNs are overparameterized. As a result, there are many possible solutions (minima) to the problem of fitting the training data. The question is which solution has better generalization, i.e., performs better on test data that the training process did not use. If we know what type of solutions has better generalizability, a related question concerns what regularization term can we use (in addition to the loss function) to nudge the system toward those more generalizable solutions. In the next two sections, we delve into these two general problems in more detail and highlight some of the most recent developments in these directions.

3.2. Stochastic Learning Dynamics: Descending Down a Fluctuating Loss Landscape. The general optimization strategy in ANNs consists of updating the weights by following the gradient of the loss function, a method called gradient descent (GD). Given the feedforward architecture of DLNNs, GD can be carried out efficiently by backpropagation. However, GD is computationally prohibitive for large datasets if one uses the overall loss function averaged over all the training data. The stochastic gradient descent (SGD) method has been used instead to circumvent the large dataset problem by updating the weights according to a subset (minibatch) of samples randomly chosen at each iteration (54, 55). Remarkably, it was subsequently discovered that SGD is also crucial for finding more generalizable solutions in DLNNs.

However, despite the tremendous successes of deep learning, the reason why SGD is so effective in learning good solutions in a high-dimensional nonconvex loss function (energy) landscape remains poorly understood. The random element seems key for SGD, yet also makes it harder to understand. Fortunately, many physical and biological systems include such random elements, e.g., Brownian motion and stochastic biochemical reactions, and powerful tools have been developed for understanding collective behaviors in stochastic systems with many degrees of freedom. Indeed, concepts and methods from statistical physics and stochastic dynamical system theory have been used recently to investigate the SGD dynamics, the loss function landscape, and their relationship in DLNNs.

To demonstrate the utility of such a physics-based approach for understanding DLNNs, we briefly describe a theoretical framework for studying SGD learning dynamics and some interesting insights gained from it. We start by considering the SGD-based learning process as a stochastic dynamical system. A learning system such as a neural network (NN) and especially a DNN has a large number (N) of weight parameters w_i ($i = 1, 2, ..., N$). For supervised learning, there is a set of M training samples each with an input $\vec{X}_k$ and a correct output $\vec{Z}_k$ for $k = 1, 2, ..., M$. For each input $\vec{X}_k$, the learning system predicts an output $\vec{Y}_k = G(\vec{X}_k, \vec{w})$, where the output function G depends on the architecture of the NN as well as its weights $\vec{w}$. The goal of learning is to find the weight parameters to minimize the difference between the predicted and correct output characterized by an overall loss function (or energy function):

$$L(\vec{w}) = M^{-1} \sum_{k=1}^{M} d(\vec{Y}_k, \vec{Z}_k), \qquad [1]$$

where $d(\vec{Y}_k, \vec{Z}_k)$ is a measure of distance between $\vec{Y}_k$ and $\vec{Z}_k$. Here, a typical distance measure is the cross-entropy.

Specifically, the change of weight w_i ($i = 1, 2, \ldots, N$) for iteration t in SGD is given by:

$$\Delta w_i(t) = -\alpha \frac{\partial L^{\mu(t)}(\vec{w})}{\partial w_i}, \qquad [2]$$

where α is the learning rate and $\mu(t)$ represents the random minibatch used for iteration t. The minibatch loss function (MLF) for minibatch μ of size B is defined as:

$$L^{\mu}(\vec{w}) = B^{-1} \sum_{l=1}^{B} d(\vec{Y}_{\mu_l}, \vec{Z}_{\mu_l}), \qquad [3]$$

where μ_l ($l = 1, 2, .., B$) labels the B randomly chosen samples.

Here, we introduce the key concept of a minibatch loss function (MLF) ensemble $\{L^{\mu}(\vec{w})\}$, i.e., an ensemble of energy landscapes each from a random minibatch. The overall loss function $L(\vec{w})$ is just the ensemble average of MLFs: $L \equiv \langle L^{\mu} \rangle_{\mu}$. The SGD noise comes from the variation between a MLF and its ensemble average: $\delta L^{\mu} \equiv L^{\mu} - L$. By taking the continuous time approximation and keeping the first-order time derivative term in Eq. **2**, we obtain the following stochastic partial differential equation for SGD:

$$\frac{\partial \vec{w}}{\partial t} = -\alpha \frac{\partial L}{\partial \vec{w}} + \vec{\eta}(\vec{w}), \qquad [4]$$

where time t and all timescales in this study are measured in the unit of minibatch iteration time $\Delta t = 1$. The continuous time limit amounts to consider time scales that are much larger than Δt, e.g., one epoch time is $M/B (\gg 1)$. Eq. **4** is analogous to the Langevin equation in statistical physics. The first term $-\alpha \frac{\partial L}{\partial \vec{w}}$ is the deterministic gradient descent governed by the overall loss function L analogous to the energy function in physics. The second term is the SGD noise term $\vec{\eta} \equiv -\alpha \nabla_{\vec{w}} \delta L^{\mu}(\vec{w})$ with zero mean $\langle \vec{\eta} \rangle = 0$ and an equal time covariance matrix

$$C_{ij}(\vec{w}) \equiv \langle \eta_i \eta_j \rangle = \alpha^2 \times \langle \frac{\partial \delta L^{\mu}}{\partial w_i} \frac{\partial \delta L^{\mu}}{\partial w_j} \rangle_{\mu}, \qquad [5]$$

which depends explicitly on $\vec{w}$, i.e., gives rise to a complex form of multiplicative noise. For a given network architecture, the learning dynamics can thus be mapped to the stochastic motion of a "learner particle" whose coordinates are the weights of the network. In particular, the SGD learning algorithm corresponds to the learner particle descending down a fluctuating energy landscape, which is governed by the Langevin equation (Eq. **2**) with a deterministic GD term and a noise term with covariance matrix given by Eq. **5**.

The most unusual and interesting part of the SGD learning dynamics comes from the noise term. As first pointed out by Chaudhauri and Soatto (56), unlike equilibrium physical systems where the noise has a constant strength given by the thermal temperature, the SGD dynamics is highly nonequilibrium as the SGD noise is neither isotropic nor homogeneous. From its definition, the SGD noise depends on the loss landscape itself. One of the most interesting findings is that the SGD noise covariance matrix is highly correlated with the Hessian matrix of the loss function: Their eigen-directions are highly aligned and the corresponding eigenvalues are highly correlated (57, 58). In particular, in sharper directions in the loss landscape (larger eigenvalues in the Hessian matrix), the SGD noise is also larger. This results in a robust inverse relation between the weight variance and the flatness of loss landscape in all directions, which is the opposite to the fluctuation–response relation (aka the Einstein relation) in equilibrium statistical physics.

There is increasing empirical evidence in support of the notion that "good" (generalizable) solutions exist at the flat (shallow) minima of the loss function (59–65); however, there is still little understanding of how SGD-based algorithms can find these flat minima in the high-dimensional weight space. The "inverse Einstein relationship" (57) obtained within the stochastic learning dynamics framework suggests that SGD serves as a landscape-dependent annealing algorithm. The effective SGD temperature decreases with the landscape flatness so the system seeks out (prefers) flat minima over sharp ones. As shown in a recent paper (58) using a Fokker–Planck equation to study the weight distribution of the SGD learning dynamics, SGD introduces a flatness-dependent term in the effective loss function that regularizes the system to prefer flatter minima.

An important class of ANN models are the generative models that are able to generate new samples by being trained on existing samples. A well-known early example is the generative adversarial network (GAN) model (66) whose learning dynamics has been studied by using a stochastic dynamical systems approach (67). Indeed, some of the most successful generative models such as the diffusion-based models (68) have their origin in physics and thus provide a rich area for physics-based research. In this special issue, Zdeborova et al. (69) make a comprehensive comparison among different generative models based on a spin-glass perspective, which sheds light on a theoretical understanding of the capabilities and limitations of these powerful generative models.

3.3. Generalization: The Blessing and Curse of High Dimensionality. Most of the problems in physics are overconstrained (or underparameterized). For example, in a protein folding problem with N amino acids, even if we only consider the pair-wise interaction energies, we have $\sim N^2$ constraints, which is much higher than the $\sim 2N$ degrees of freedom, i.e., the independent coordinates of amino acids on a 1D chain. Typically, an overconstrained problem has a unique solution. This situation is illustrated in Fig. 3*A* where the energy landscape has a unique minimum, which corresponds to the native structure of the folded protein. Solving the overconstrained problem by minimizing the overall energy function, e.g., ab initio protein folding is a notoriously hard problem. On the other hand, DLNNs are overparameterized. The number of parameters (weights) is much larger than

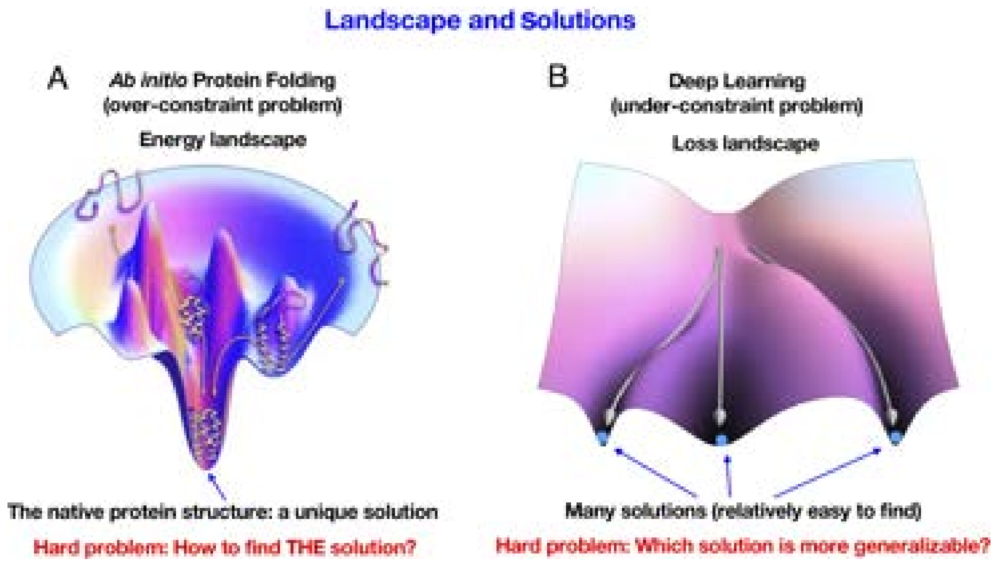
Landscape and Solutions
A
Ab initio Protein Folding
(over-constraint problem)
Energy landscape
The native protein structure: a unique solution
Hard problem: How to find THE solution?
B
Deep Learning
(under-constraint problem)
Loss landscape
Many solutions (relatively easy to find)
Hard problem: Which solution is more generalizable?

Fig. 3. The difference of landscape and solutions in overconstrained and underconstrained problems. (*A*) The free energy landscape in protein folding (an overconstrained problem) where there is a unique global minimum, which is typically very hard to find. The image is adopted from Dill and Maccallum (70). (*B*) The loss landscape in an overparameterized (underconstrained) deep learning model, which can have many global minima. The challenge is to find out which solution is more generalizable.

the internal degrees of freedom in the data. The blessing of having a large number of parameters in DLNNs is that it makes finding a solution (a minimum in the loss landscape) relatively easy. However, the curse of high dimensionality in the parameter space is that there are many solutions (minima of the loss function), as shown in Fig. 3*B*. Thus, the important question becomes which one of the many solutions performs better for the test data, i.e., which solution has better generalizability.

Indeed, generalization is one of the most important problems in machine learning. This problem becomes more pressing given the overwhelming number of parameters (weights) used in DLNNs. There has been much work on generalization in DLNN based on various theoretically and empirically motivated complexity measures (VC-dimension, norm of parameters, sharpness, path norms, etc.). As summarized in a recent review by Jiang et al. (71), although empirical evidence suggests a strong correlation between sharpness-based measures and generalization (72), many other (theoretically motivated) measures such as the norm-based measures do not serve as robust indicators for generalization (71). Even for the sharpness-based measures, we do not understand why and how they are effective in predicting generalization. Furthermore, there is recent work challenging the validity of using loss landscape sharpness alone for determining generalization, based on a general scaling invariance in DLNN as pointed out by Dinh et al. (73). Indeed, a comprehensive understanding of generalization in DLNN remains elusive.

One key question in generalization is what properties of a given solution determine its generalizability. The difficulty in answering this problem is that while learning is guided by the training loss, the generalization performance is evaluated by the testing loss, and it is difficult to make theoretical progress without access to the test loss landscape. Recently, this problem was tackled by using an equivalence (duality) between changes in data and changes in weight parameters (74). The general idea is that if the change of the input between a training data ($\boldsymbol{x}$) and a testing data ($\boldsymbol{x'}$) is equivalent to a corresponding change of the weights from the solution ($\boldsymbol{W}$) to a new weight ($\boldsymbol{W'}$), we can then use this duality relation to map a distribution in the input space to a distribution in the weight space, where we can evaluate the generalization loss. Remarkably, an infinite family of such exact duality relations in any densely connected layer has been found. By using the "minimal" duality relation with the smallest weight change, the generalization loss can be decomposed to contributions from different eigen-directions of the Hessian matrix of the loss function at the solution in the weight space. The form of these contributions reveals two distinct factors (determinants) for generalization—one is governed by the sharpness of the loss landscape and the other corresponds to the norm of the solution weighted by the covariance of the relative differences between the training and testing data. One of the main insights gained from this study is that these two determinants multiply together to determine generalization, which resolves the puzzle regarding flatness raised by Dinh et al. (73).

In underconstrained (or overparameterized) learning systems such as DLNNs, regularization is an important component added to the loss function in order to push the system toward those solutions that have higher generalizability. Despite its importance, however, regularizations are typically based on some intuitions of what properties of a more generalizable solution should have. From the perspective of having two contributing factors (sharpness and size) affecting the generalization loss, the mechanisms behind the ability of SGD and weight decay as two effective regulation schemes become clear. Clearly, designing regularization schemes based on properties of the underlying system (e.g., symmetry and conservation law in physical systems) and/or certain general factors affecting the generalizability of a solution would make an interesting future research direction.

As an extreme case of overfitting, DLNN can "memorize" all the training samples even if their labels are replaced by pure noise (75). Such overfitted (memorization) solution has no generalizability. Remarkably, DLNN avoids overfitting and testing error follows the so called "double descent" curve (76). As the model capacity (complexity) increases, the test error follows the usual U-shaped curve at the beginning, first decreasing and then peaking near the interpolation threshold when the model achieves vanishing training error. However, it descends again as model capacity exceeds this interpolation threshold, with the test error reaching its (global) minimum in the overparameterization regime where the number of parameters is much larger than the number of samples. Rapid progress has been made in understanding this double descent behavior by using simple models. For example, both optimization and generalization guarantees for overparameterized simple two-layer networks are proven with leaky ReLU activation on linearly separable data (77). This result has subsequently been extended to 2-layer networks with ReLU activation (78) and 2&3-layer networks with smooth activation functions (79). In a different approach by using the Neural Tangent Kernel (80), which connects large (wide) neural nets to kernel methods, it was shown that the generalization error decreases toward a plateau value in a power-law fashion as $N_p^{-1/2}$ with N_p the number of parameters in the overparameterized regime (81). In simple synthetic learning models such as the random features model with ridge regression loss function, the double descent behavior has been shown analytically (82). This analytical result has been extended to other synthetic learning models (e.g., the random manifold model) and for more general loss functions by using the replica method (83).

In fact, one of the most exciting empirical findings in large models such as large language models (LLMs) is that the generalization loss keeps decreasing with an apparent power-law dependence on the model size and the data size when they increase together in a proportional fashion. Physicists are naturally drawn to behaviors described by power-laws and have developed powerful tools such as renormalization group theory to explain scaling laws in critical phenomena. Therefore, we believe that understanding the "power law" dependence of generalization on data size and model size in large complex learning systems is one of the most tantalizing and highly important directions to pursue for physicists. In this special issue, Bahri et al (84) investigated the possible origins behind such "scaling laws" and provided a taxonomy for different scaling regimes.

3.4. Inspiration from Realistic Neural Networks and Real Neurons. As we stated above, artificial neural networks benefited from two natural science disciplines, namely neuroscience and statistical physics. However, other than the initial inspiration drawn from neuroscience that is embodied in the McCulloch–Pitts neurons and the layered feedforward neural network (perceptron) architecture, there has not been much neuroscience insight incorporated in DLNNs. Even though this special issue is mostly about the cross talk between physics and machine learning, the need for novel concepts arising from neuroscience is greater than ever. There are several specific limiting architectural aspects of DLNN that could benefit from a stronger grounding in neuroscience principles. For example, the successes of deep learning have been limited mostly to static tasks with static datasets and furthermore require huge amounts of explicitly labeled data. Since many researchers have noted that biological brains are exquisitely adapted to dynamic tasks in dynamic environments, we think it is plausible that a better understanding of how brains perform dynamic tasks will lead to new concepts that will drive improvements in ML performance on such tasks. Novel brain-inspired algorithms could come from exploring major differences between actual computations in brains and DLNN algorithms and architectures. In this special issue, Haim Sompolinksy et al. (85) present a novel perspective and in-depth comparison between artificial and brain neural networks in terms of their representations and generalization.

In addition to representations and generalization, we list two other differences between artificial and brain networks with the hope of stimulating future work as they can be both studied with the physics-based approach outlined in the previous sections:

- Brains learn with a local learning rule and little supervision. First and foremost, DLNNs focus mostly on supervised learning, where there are explicit labels denoting the correct output for a given input pattern, whereas brains appear to do very little supervised learning. Instead, theory and experimental data suggest that neural learning employs mostly unsupervised, temporal-predictive, and reinforcement learning (RL) techniques. At the algorithmic level, learning in DLNNs is carried out by backpropagation, which is a global learning rule, whereas learning in brains is achieved by local learning rules such as the Hebbian rule.
- Brains are highly dynamic and constantly interact with environments. Most DLNNs use static feed-forward architectures, or they have relaxational properties leading to stationary states. By contrast, brains exhibit complex dynamic behavior (e.g., different brain rhythms/oscillations) enabled by massive recurrent connections. Furthermore, current DLNNs are almost exclusively devoted to static perception-only tasks, whereas the overarching purpose of the brain is to generate behavior in a continual perception-action loop with the environment.

Novel inspiration from neuroscience can also be gained at the single neuron level as presented in this special issue by Chklovskii et al. (86). The authors introduce the concept of neurons as feedback controllers of their environments, a role far beyond the traditional McCulloch–Pitts neurons. This innovative approach not only explains various experimental findings that previously seemed unrelated, it may also point the way for the creation of more sophisticated, biologically inspired artificial intelligence systems.

ACKNOWLEDGMENTS. We thank David Hathcock for providing Fig. 3*B*. HL would like to acknowledge discussions with Jose Onuchic on protein folding. H.L. was supported in part by the Center for Theoretical Biological Physics, NSF PHY-2019745. Y.T. would like to thank the Center for Computational Biology at the Flatiron Institute for hospitality while a portion of this work was carried out.

1. J. Jumper *et al.*, Highly accurate protein structure prediction with alphafold. *Nature* **596**, 583–589 (2021).
2. K. A. Dill, J. L. MacCallum, The protein-folding problem, 50 years on. *Science* **338**, 1042–1046 (2012).
3. C. M. Dobson, Protein folding and misfolding. *Nature* **426**, 884–890 (2003).
4. J. N. Onuchic, P. G. Wolynes, Theory of protein folding. *Curr. Opin. Struct. Biol.* **14**, 70–75 (2004).
5. M. K. Higgins, Can we alphafold our way out of the next pandemic? *J. Mol. Biol.* **433**, 167093 (2021).
6. H. Park, P. Patel, R. Haas, E. Huerta, APACE: Alphafold2 and advanced computing as a service for accelerated discovery in biophysics. *Proc. Natl. Acad. Sci. U.S.A.* **121**, e2311888121 (2024).
7. N. Qian, T. J. Sejnowski, Predicting the secondary structure of globular proteins using neural network models. *J. Mol. Biol.* **202**, 865–884 (1988).
8. M. Minsky, S. A. Papert, *Perceptrons, Reissue of the 1988 Expanded Edition with a New Foreword by Léon Bottou: An Introduction to Computational Geometry* (MIT Press, 2017).
9. D. H. Ackley, G. E. Hinton, T. J. Sejnowski, A learning algorithm for Boltzmann machines. *Cognit. Sci.* **9**, 147–169 (1985).
10. D. E. Rumelhart, J. L. McClelland, *Corporate PDP Research Group, Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 1: Foundations* (MIT Press, 1986).
11. Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition. *Proc. IEEE* **86**, 2278–2324 (1998).
12. P. Baldi, "Autoencoders, unsupervised learning, and deep architectures" in *Proceedings of ICML Workshop on Unsupervised and Transfer Learning* (JMLR Workshop and Conference Proceedings, 2012), pp. 37–49.
13. J. Gui, Z. Sun, Y. Wen, D. Tao, J. Ye, A review on generative adversarial networks: Algorithms, theory, and applications. *IEEE Trans. Knowl. Data Eng.* **35**, 3313–3332 (2021).
14. J. J. Hopfield, Neural networks and physical systems with emergent collective computational abilities. *Proc. Natl. Acad. Sci. U.S.A.* **79**, 2554–2558 (1982).
15. J. Martin, M. Lequerica-Mateos, J. Onuchic, I. Coluzza, F. Morcoc, Machine learning in biological physics: From biomolecular prediction to design. *Proc. Natl. Acad. Sci. U.S.A.* **121**, e2311807121 (2024).
16. Protein data bank, The single global archive for 3D macromolecular structure data. *Nucleic Acids Res.* **47**, D520–D528 (2019).
17. J. I. Sułkowska, F. Morcos, M. Weigt, T. Hwa, J. N. Onuchic, Genomics-aided structure prediction. *Proc. Natl. Acad. Sci. U.S.A.* **109**, 10340–10345 (2012).
18. D. De Juan, F. Pazos, A. Valencia, Emerging methods in protein co-evolution. *Nat. Rev. Genet.* **14**, 249–261 (2013).
19. A. Davtyan *et al.*, AWSEM-MD: Protein structure prediction using coarse-grained physical potentials and bioinformatically based local structure biasing. *J. Phys. Chem. B* **116**, 8494–8503 (2012).
20. J. D. Bryngelson, P. G. Wolynes, Spin glasses and the statistical mechanics of protein folding. *Proc. Natl. Acad. Sci. U.S.A.* **84**, 7524–7528 (1987).
21. J. D. Bryngelson, J. N. Onuchic, N. D. Socci, P. G. Wolynes, Funnels, pathways, and the energy landscape of protein folding: A synthesis. *Prot.: Struct. Funct. Bioinf.* **21**, 167–195 (1995).
22. S. Yang *et al.*, Domain swapping is a consequence of minimal frustration. *Proc. Natl. Acad. Sci. U.S.A.* **101**, 13786–13791 (2004).
23. R. D. Hills Jr, C. L. Brooks III, Insights from coarse-grained gō models for protein folding and dynamics. *Int. J. Mol. Sci.* **10**, 889–905 (2009).
24. S. Tripathi, D. A. Kessler, H. Levine, Biological networks regulating cell fate choice are minimally frustrated. *Phys. Rev. Lett.* **125**, 088101 (2020).
25. K. M. Ruff, R. V. Pappu, Alphafold and implications for intrinsically disordered proteins. *J. Mol. Biol.* **433**, 167208 (2021).
26. M. Di Pierro, B. Zhang, E. L. Aiden, P. G. Wolynes, J. N. Onuchic, Transferable model for chromosome architecture. *Proc. Natl. Acad. Sci. U.S.A.* **113**, 12168–12173 (2016).
27. M. A. Marti-Renom, L. A. Mirny, Bridging the resolution gap in structural modeling of 3D genome organization. *PLoS Comput. Biol.* **7**, e1002125 (2011).
28. U. Lupo, D. Sgarbossa, A. F. Bitbol, Pairing interacting protein sequences using masked language modeling. *Proc. Natl. Acad. Sci. U.S.A.* **121**, e2311887121 (2024).
29. B. Meynard-Piganeau, C. Feinauer, M. Weigt, A. M. Walczak, T. Mora, Tulip-a transformer based unsupervised language model for interacting peptides and T-cell receptors that generalizes to unseen epitopes. bioRxiv [Preprint] (2023). https://www.biorxiv.org/content/10.1101/2023.07.19.549669v1 (Accessed 10 January 2024).
30. B. P. Kwee *et al.*, STAPLER: Efficient learning of TCR-peptide specificity prediction from full-length TCR-peptide data. bioRxiv [Preprint] (2023). https://www.biorxiv.org/content/10.1101/2023.04.25.538237v1 (Accessed 10 January 2024).
31. A. T. Wang *et al.*, RACER-m leverages structural features for sparse T cell specificity prediction. bioRxiv [Preprint] (2023). https://www.biorxiv.org/content/10.1101/2023.08.06.552190v1 (Accessed 3 January 2024).
32. B. A. Camley, W. J. Rappel, Physical models of collective cell motility: From cell to tissue. *J. Phys. D: Appl. Phys.* **50**, 113002 (2017).
33. M. Basan, J. Elgeti, E. Hannezo, W. J. Rappel, H. Levine, Alignment of cellular motility forces with tissue flow as a mechanism for efficient wound healing. *Proc. Natl. Acad. Sci. U.S.A.* **110**, 2452–2459 (2013).
34. V. Hakim, P. Silberzan, Collective cell migration: A physics perspective. *Rep. Progr. Phys.* **80**, 076601 (2017).
35. J. LaChance, K. Suh, J. Clausen, D. J. Cohen, Learning the rules of collective cell migration using deep attention networks. *PLoS Comput. Biol.* **18**, e1009293 (2022).
36. S. U. Hirway, S. H. Weinberg, A review of computational modeling, machine learning and image analysis in cancer metastasis dynamics. *Comput. Syst. Oncol.* **3**, e1044 (2023).
37. S. Al-Janabi, A. Huisman, P. J. Van Diest, Digital pathology: Current status and future perspectives. *Histopathology* **61**, 1–9 (2012).
38. D. B. Brückner *et al.*, Stochastic nonlinear dynamics of confined cell migration in two-state systems. *Nat. Phys.* **15**, 595–601 (2019).
39. R. Yu, R. Wang, Learning dynamical systems from data: An introduction to physics-guided deep learning. *Proc. Natl. Acad. Sci. U.S.A.* **121**, e2311808121 (2024).
40. D. Kochkov *et al.*, Machine learning-accelerated computational fluid dynamics. *Proc. Natl. Acad. Sci. U.S.A.* **118**, e2101784118 (2021).
41. E. M. King, C. X. Du, Q.-Z. Zhu, S. S. Schoenholz, M. P. Brenner, Programming patchy particles for materials assembly design. *Proc. Natl. Acad. Sci. U.S.A.* **121**, e2311891121 (2024).
42. R. E. Wengert, A simple automatic derivative evaluation program. *Commun. ACM* **7**, 463–464 (1964).
43. F. Ruehle, Data science applications to string theory. *Phys. Rep.* **839**, 1–117 (2020).
44. Y. LeCun, Y. Bengio, G. Hinton, Deep learning. *Nature* **521**, 436 EP (2015).
45. I. Goodfellow, A. Courville, Y. Bengio, *Deep Learning* (MIT Press, 2016), vol. 1.
46. K. He, X. Zhang, S. Ren, J. Sun, "Deep residual learning for image recognition" in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2016), pp. 770–778.
47. Y. Wu *et al.*, Google's neural machine translation system: Bridging the gap between human and machine translation. arXiv [Preprint] (2016). http://arxiv.org/abs/1609.08144 (Accessed 3 January 2024).
48. D. Silver *et al.*, Mastering the game of go with deep neural networks and tree search. *Nature* **529**, 484–489 (2016).
49. W. Mcculloch, W. Pitts, A logical calculus of ideas immanent in nervous activity. *Bull. Math. Biophys.* **5**, 127–147 (1943).
50. D. J. Amit, H. Gutfreund, H. Sompolinsky, Spin-glass models of neural networks. *Phys. Rev. A* **32**, 1007 (1985).
51. F. Rosenblatt, The perceptron: A probabilistic model for information storage and organization in the brain. *Psychol. Rev.* **65**, 386–408 (1958).
52. P. W. Anderson, More is different. *Science* **177**, 393–396 (1972).
53. S. Ambrose, M. Bridges, M. Lovett, *How Learning Works: 7 Research-Based Principles for Smart Teaching* (John Wiley and Sons, San Francisco, 2010).
54. H. Robbins, S. Monro, A stochastic approximation method. *Ann. Math. Stat.* **22**, 400–407 (1951).
55. L. Bottou, "Large-scale machine learning with stochastic gradient descent" in *Proceedings of COMPSTAT 2010*, Y. Lechevallier, G. Saporta Eds. (Physica-Verlag HD, Heidelberg, 2010), pp. 177–186.
56. P. Chaudhari, S. Soatto, "Stochastic gradient descent performs variational inference, converges to limit cycles for deep networks" in *2018 Information Theory and Applications Workshop (ITA)* (2018). http://dx.doi.org/10.1109/ita.2018.8503224.
57. Y. Feng, Y. Tu, The inverse variance-flatness relation in stochastic gradient descent is critical for finding flat minima. *Proc. Natl. Acad. Sci. U.S.A.* **118** (2021).
58. N. Yang, C. Tang, Y. Tu, Stochastic gradient descent introduces an effective landscape-dependent regularization favoring flat solutions. *Phys. Rev. Lett.* **130**, 237101 (2023).
59. G. E. Hinton, D. van Camp, "Keeping the neural networks simple by minimizing the description length of the weights" in *Proceedings of the Sixth Annual Conference on Computational Learning Theory, COLT 1993* (ACM, New York, NY, USA, 1993), pp. 5–13.
60. S. Hochreiter, J. Schmidhuber, Flat minima. *Neural Comput.* **9**, 1–42 (1997).
61. C. Baldassi *et al.*, Unreasonable effectiveness of learning neural networks: From accessible states and robust ensembles to basic algorithmic schemes. *Proc. Natl. Acad. Sci. U.S.A.* **113**, E7655–E7662 (2016).
62. P. Chaudhari *et al.*, *Entropy-SGD: Biasing Gradient Descent into Wide Valleys* (ICLR, 2017).
63. Y. Zhang, A. M. Saxe, M. S. Advani, A. A. Lee, Energy-entropy competition and the effectiveness of stochastic gradient descent in machine learning. *Mol. Phys.* **116**, 3214–3223 (2018).
64. S. Mei, A. Montanari, P. M. Nguyen, A mean field view of the landscape of two-layer neural networks. *Proc. Natl. Acad. Sci. U.S.A.* **115**, E7665–E7671 (2018).
65. C. Baldassi, F. Pittorino, R. Zecchina, Shaping the learning landscape in neural networks around wide flat minima. *Proc. Natl. Acad. Sci. U.S.A.* **117**, 161–170 (2020).
66. I. Goodfellow *et al.*, "Generative adversarial nets" in *Advances in Neural Information Processing Systems*, Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, K. Weinberger, Eds. (Curran Associates, Inc., 2014), vol. 27.
67. S. Durr, Y. Mroueh, Y. Tu, S. Wang, Effective dynamics of generative adversarial networks. *Phys. Rev. X* **13**, 041004 (2023).
68. J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, S. Ganguli, "Deep unsupervised learning using nonequilibrium thermodynamics" in *Proceedings of the 32nd International Conference on Machine Learning, Proceedings of Machine Learning Research*, F. Bach, D. Blei, Eds. (PMLR, Lille, France, 2015), vol. 37, pp. 2256–2265.
69. D. Ghioa, Y. Dandi, F. Krzakala, L. Zdeborova, Sampling with flows, diffusion and autoregressive neural networks from a spin-glass perspective. *Proc. Natl. Acad. Sci. U.S.A.* **121**, e2311810121 (2024).
70. K. Dill, J. Maccallum, *The Protein-Folding Problem, 50 Years on* (Science New York, N.Y., 2012), vol. 338, pp. 1042–1046.
71. Y. Jiang, B. Neyshabur, H. Mobahi, D. Krishnan, S. Bengio, Fantastic generalization measures and where to find them. *ICLR* (2020).
72. N. S. Keskar, D. Mudigere, J. Nocedal, M. Smelyanskiy , P. T. P. Tang , On large-batch training for deep learning: Generalization gap and sharp minima. *ICLR* (2017).
73. L. Dinh, R. Pascanu, S. Bengio, Y. Bengio, "Sharp minima can generalize for deep nets" in *Proceedings of 34th International Conference Machine Learning* (2017), vol. 70, pp. 1019–1028.
74. Y. Feng, W. Zhang, Y. Tu, Activity-weight duality in feed-forward neural networks reveals two co-determinants for generalization. *Nat. Mach. Intell.* **5**, 908–918 (2023).
75. C. Zhang, S. Bengio, M. Hardt, B. Recht, O. Vinyals, Understanding deep learning requires rethinking generalization. *ICLR* (2017).
76. M. Belkin, D. Hsu, S. Ma, S. Mandal, Reconciling modern machine-learning practice and the classical bias–variance trade-off. *Proc. Natl. Acad. Sci. U.S.A.* **116**, 15849–15854 (2019).
77. A. Brutzkus, A. Globerson, E. Malach , S. Shalev-Shwartz , SGD learns over-parameterized networks that provably generalize on linearly separable data. *ICLR* (2018).
78. Y. Li, Y. Liang, Learning overparameterized neural networks via stochastic gradient descent on structured data. *Adv. Neural Inf. Process. Syst.* **31**, 8157–8166 (2018).
79. Z. Allen-Zhu, Y. Li, Z. Song, "A convergence theory for deep learning via over-parameterization" in *International Conference Machine Learning* (2019), pp. 242–252.
80. A. Jacot, F. Gabriel, C. Hongler, Neural tangent kernel: Convergence and generalization in neural networks. *Adv. Neural Inf. Process. Syst.* **31**, 8571–8580 (2018).
81. M. Geiger *et al.*, Scaling description of generalization with number of parameters in deep learning. *J. Stat. Mech.: Theory Exp.* **2020**, 023401 (2020).

82. S. Mei, A. Montanari, The generalization error of random features regression: Precise asymptotics and the double descent curve. *Commun. Pure Appl. Math.* **75**, 667–766 (2022).
83. F. Gerace, B. Loureiro, F. Krzakala, M. Mézard, L. Zdeborová, Generalisation error in learning with random features and the hidden manifold model (ICML, 2020), pp. 3452–3462.
84. Y. Bahri, E. Dyer, J. Kaplan, J. Lee, U. Sharma, Explaining neural scaling laws. *Proc. Natl. Acad. Sci.* **121**, e2311878121 (2024).
85. Q. Li, B. Sorscher, H. Sompolinsky, Representations and generalization in artificial and brain neural networks. *Proc. Natl. Acad. Sci. U.S.A.* **121**, e2311805121 (2024).
86. J. Moore *et al*., The neuron as a direct data-driven controller. *Proc. Natl. Acad. Sci. U.S.A.* 2023–11893 (2024).