Predicting the outputs of finite deep neural networks trained with noisy gradients

Gadi Naveh [1,2,*] Oded Ben David [3] Haim Sompolinsky,[1,2,4] and Zohar Ringel[1]

[1]*Racah Institute of Physics, Hebrew University, Jerusalem 91904, Israel*
[2]*Edmond and Lily Safra Center for Brain Sciences, Hebrew University, Jerusalem 91904, Israel*
[3]*Agilent Research Laboratories, Israel 8 Kishon st, Bnei-Brak 5120308, Petach Tikva, Israel*
[4]*Center for Brain Science, Harvard University, Cambridge, Massachusetts 02138, USA*

A recent line of works studied wide deep neural networks (DNNs) by approximating them as Gaussian processes (GPs). A DNN trained with gradient flow was shown to map to a GP governed by the neural tangent kernel (NTK), whereas earlier works showed that a DNN with an i.i.d. prior over its weights maps to the so-called neural network Gaussian process (NNGP). Here we consider a DNN training protocol, involving noise, weight decay, and finite width, whose outcome corresponds to a certain non-Gaussian stochastic process. An analytical framework is then introduced to analyze this non-Gaussian process, whose deviation from a GP is controlled by the finite width. Our contribution is threefold: (i) In the infinite width limit, we establish a correspondence between DNNs trained with noisy gradients and the NNGP, not the NTK. (ii) We provide a general analytical form for the finite width correction (FWC) for DNNs with arbitrary activation functions and depth and use it to predict the outputs of empirical finite networks with high accuracy. Analyzing the FWC behavior as a function of n, the training set size, we find that it is negligible for both the very small n regime, and, surprisingly, for the large n regime [where the GP error scales as $O(1/n)$]. (iii) We flesh out algebraically how these FWCs can improve the performance of finite convolutional neural networks (CNNs) relative to their GP counterparts on image classification tasks.

DOI: 10.1103/PhysRevE.104.064301

I. INTRODUCTION

Deep neural networks (DNNs) have been rapidly advancing the state-of-the-art in machine learning, yet a complete analytic theory remains elusive. Recently, several exact results were obtained in the highly over-parameterized regime [$N \to \infty$, where N denotes the width or number of channels for fully connected networks (FCNs) and convolutional neural networks (CNNs), respectively] [1]. This facilitated the derivation of an exact correspondence with Gaussian processes (GPs) known as the neural tangent kernel (NTK) [2]. The latter holds when highly over-parameterized DNNs are trained by gradient flow, namely, with vanishing learning rate and involving no stochasticity.

The NTK result has provided the first example of a DNN to GP correspondence valid after end-to-end DNN training. This theoretical breakthrough allows one to think of DNNs as inference problems with underlying GPs [3]. For instance, it provides a quantitative description of the generalization properties [4,5] and training dynamics [2,6] of DNNs.

Despite its novelty and importance, the NTK correspondence suffers from a few shortcomings: (a) In the NTK parametrization, the weights of the network change only slightly from their initial random values, thus it is a form of "lazy learning" [7], unlike finite DNNs which are typically in the *feature learning* regime [8]. (b) The deterministic

training of NTK is qualitatively different from the stochastic one used in practice. (c) NTK typically under-performs convolutional neural networks (CNNs) trained with stochastic gradient descent (SGD) [9] on real-world tasks such as image classification tasks. (d) Deriving explicit finite width corrections (FWCs) is challenging, as it requires solving a set of coupled ODEs [10,11]. Thus, there is a need for an extended theory for end-to-end trained deep networks which is valid for finite width DNNs.

Our contribution is threefold. First, we establish a correspondence between a DNN trained with noisy gradients and a statistical field theory description of the distribution of the DNN's outputs (Sec. II). We show that this stochastic process (SP) converges to the neural network Gaussian process (NNGP) as $N \to \infty$. In previous works on the NNGP correspondence [12,13] the NNGP kernel is determined by the distribution of the DNN weights *at initialization*, whereas in our correspondence the weights are sampled across the stochastic training dynamics, drifting far away from their initial values and acquiring nontrivial statistics. We call our correspondence the *neural network stochastic process (NNSP)*, and show that it holds when the training dynamics in output space exhibit ergodicity, which we validate numerically in several different settings.

Second, we predict the outputs of trained finite-width DNNs, significantly improving upon the corresponding GP predictions (Sec. III). This is done by a perturbation theory derivation of leading FWCs which are found to scale with width as $1/N$. The accuracy at which we can predict the

*gad.mintz@mail.huji.ac.il
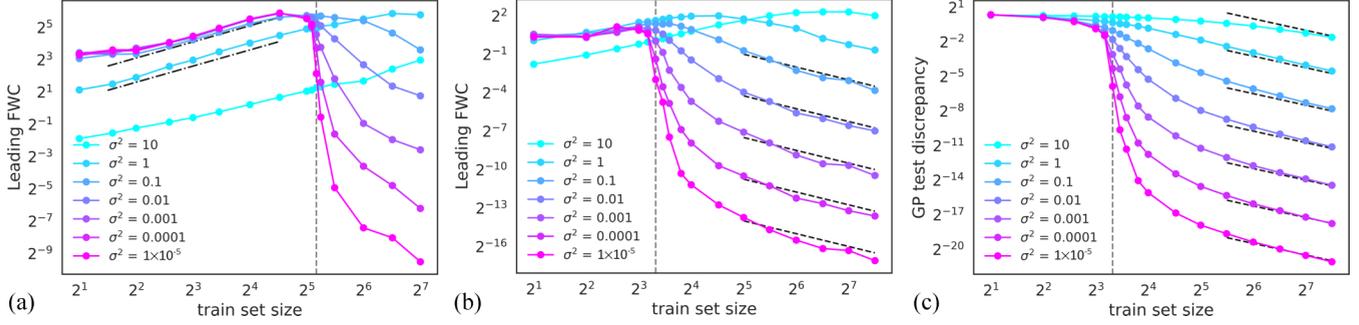
FIG. 1. Leading FWC to the mean $|\bar{f}_U(x_*)|$ [Eq. (11)] and GP discrepancy (RMSE) as a function of train set size n for varying training noise σ^2. The target is quadratic $g(x) = x^\mathsf{T} A x = O(1)$ with $x \in \mathbb{S}_{d-1}(\sqrt{d})$ so the number of parameters to be learnt is $d(d+1)/2$ (vertical gray dashed line). The GP discrepancy is monotonically decreasing with n whereas $|\bar{f}_U(x_*)|$ increases linearly for small n [dashed-dotted lines in panel (a)] before it decays (best illustrated for larger d and σ^2). For sufficiently large n, both the GP discrepancy and $|\bar{f}_U(x_*)|$ scale as $1/n$ [diagonal dashed black lines in panels (b), (c)]. This verifies our prediction for the scaling of FWCs with n, Eq. (14) in the large n regime. Notably, it implies that at large N FWCs are only important at intermediate values of n.

empirical DNNs' outputs serves as a strong verification for our aforementioned ergodicity assumption. In the regime where the GP RMSE scales as $1/n$, we find that the leading FWC is a decaying function of n, and thus overall negligible. In the small n regime we find that the FWC is small and grows with n (Fig. 1).

Third, our formalism sheds light on the empirical observation that infinite channel CNNs (with no pooling layers) do not benefit from weight sharing, which is present in finite CNNs [14]. We show that the leading correction to the GP limit already captures weight sharing effects. Alongside this, we validate that the NNSP correspondence holds also when training CNNs (Sec. IV B).

Overall, the NNSP correspondence provides a rich analytical and numerical framework for exploring the theory of deep learning, unique in its ability to incorporate finite overparameterization, stochasticity, and depth.

Related work

The idea of leveraging the dynamics of the gradient descent algorithm for approximating Bayesian inference has been considered in various works [15–19]. However, to the best of our knowledge, a correspondence with a concrete SP or a nonparametric model was not established nor was a comparison made of the DNN's outputs with analytical predictions.

Finite width corrections were studied recently in the context of the NTK correspondence by several authors. Hanin and Nica [20] studied the NTK of finite DNNs, but where the depth scales together with width, whereas we keep the depth fixed. Dyer and Gur-Ari [10] obtained a finite N correction to the linear integral equation governing the evolution of the predictions on the training set. Our work differs in several aspects: (a) We describe a different correspondence under different a training protocol with qualitatively different behavior. (b) We derive relatively simple formulas for the outputs which become entirely explicit at large n. (c) We account for all sources of finite N corrections whereas finite N NTK randomness remained an empirical source of corrections not accounted for by Dyer and Gur-Ari [10]. (d) Our formalism differs considerably: its statistical mechanical nature enables

one to import various standard tools for treating randomness, ergodicity breaking, and taking into account nonperturbative effects. (e) We have no smoothness limitation on our activation functions and provide FWCs on a generic data point and not just on the training set.

Another recent paper [21] studied Bayesian inference with weakly non-Gaussian priors induced by finite-N DNNs. Unlike here, there was no attempt to establish a correspondence with *trained* DNNs. The formulation presented here has the conceptual advantage of representing a distribution over *function space* for arbitrary training and test data, rather than over specific draws of data sets. This is useful for studying the large n behavior of learning curves, where analytical insights into generalization can be gained [4]. Last, we further find novel expressions for the fourth cumulant for ReLU activation for four randomly chosen points.

A somewhat related line of work studied the mean-field regime of shallow NNs [22–24]. We point out the main differences from our work: (a) The NN output is scaled differently with width. (b) In the mean-field regime one is interested in the dynamics (finite t) of the distribution over the NN parameters in the form of a PDE of the Fokker-Planck type. In contrast, in our framework we are interested in the distribution over function space at equilibrium, i.e., for $t \to \infty$. (c) The mean-field analysis was originally tailored for two-layer fully connected NNs and is challenging to generalize to deeper architectures [25,26] or to CNNs. In contrast, our formalism generalizes to deeper fully connected NNs and to CNNs as well, as we show in Sec. IV B.

II. THE NNSP CORRESPONDENCE

In this section we show that finite-width DNNs, trained in a specific manner, correspond to Bayesian inference using a nonparametric model which tends to the NNGP as $N \to \infty$. We first give a short review of Langevin dynamics in weight space as described by Neal *et al.* [27], Welling and Teh [18], which we use to generate samples from the posterior over weights. We then shift our perspective and consider the corresponding distribution over functions induced by the DNN, which characterizes the nonparametric model.

A. Recap of Langevin-type dynamics

Consider a DNN trained with *full-batch* gradient descent while injecting white Gaussian noise and including a weight decay term, so that the discrete time dynamics of the weights read

$$\Delta w_t := w_{t+1} - w_t = -[\gamma w_t + \nabla_w \mathcal{L}(z_w)]\eta + \sqrt{2T\eta}\xi_t, \quad (1)$$

where w_t is the vector of all network weights at time step t, γ is the strength of the weight decay, $\mathcal{L}(z_w)$ is the loss as a function of the output z_w, T is the temperature (the magnitude of noise), η is the learning rate and $\xi_t \sim \mathcal{N}(0, I)$. As $\eta \to 0$ these discrete-time dynamics converge to the continuous-time Langevin equation given by $\dot{w}(t) = -\nabla_w[\frac{\gamma}{2}||w(t)||^2 + \mathcal{L}(z_w)] + \sqrt{2T}\xi(t)$ with $\langle \xi_i(t)\xi_j(t') \rangle = \delta_{ij}\delta(t - t')$, so that as $t \to \infty$ the weights will be sampled from the equilibrium Gibbs distribution in weight space, given by [28]

$$P(w) \propto \exp\left\{-\frac{1}{T}\left[\frac{\gamma}{2}||w||^2 + \mathcal{L}(z_w)\right]\right\}$$

$$= \exp\left\{-\left[\frac{1}{2\sigma_w^2}||w||^2 + \frac{1}{2\sigma^2}\mathcal{L}(z_w)\right]\right\}. \quad (2)$$

The above equality holds since the equilibrium Gibbs distribution of the Langevin dynamics is also the posterior distribution of a Bayesian neural network (BNN) with an i.i.d. Gaussian prior on the weights $w \sim \mathcal{N}(0, \sigma_w^2 I)$. Thus, we can map the hyperparameters of the training to those of the BNN: $\sigma_w^2 = T/\gamma$ and $\sigma^2 = T/2$. Notice that a sensible scaling for the weight variance at layer ℓ is $\sigma_{w,\ell}^2 \sim O(1/N_{\ell-1})$, thus the weight decay needs to scale as $\gamma_\ell \sim O(N_{\ell-1})$. We choose to scale γ and not T since we would like to keep σ^2 fixed.

B. A transition from weight space to function space

We aim to move from a distribution over weight space as in Eq. (2), to one over function space. Namely, we consider the distribution of $z_w(x)$ implied by the above $P(w)$ where for concreteness we consider a DNN with a single scalar output $z_w(x) \in \mathbb{R}$ on a regression task with data $\{(x_\alpha, y_\alpha)\}_{\alpha=1}^n \subset \mathbb{R}^d \times \mathbb{R}$. Denoting by $P[f]$ the induced measure on function space we formally write

$$P[f] = \int dw \delta[f - z_w]P(w)$$

$$\propto e^{-\frac{1}{2\sigma^2}\mathcal{L}[f]} \int dw e^{-\frac{1}{2\sigma_w^2}||w||^2}\delta[f - z_w], \quad (3)$$

where $\int dw$ denotes an integral over all weights and we denote by $\delta[f - z_w]$ a δ function in function-space. As common in path-integrals or field-theory formalism [29], such a δ function is understood as a limit procedure where one chooses a suitable basis for function space, trims it to a finite subset, treats $\delta[f - z_w]$ as a product of regular δ functions, and finally takes the size of the subset to infinity.

Notice that the posterior over functions Eq. (3) is naturally decomposed into a likelihood term $e^{-\frac{1}{2\sigma^2}\mathcal{L}[f]}$ and a prior over functions $P_0[f] \propto \int dw e^{-\frac{1}{2\sigma_w^2}||w||^2}\delta[f - z_w]$. All the information about the network's inductive bias is encoded in the prior $P_0(f)$, since the likelihood is independent of the network

architecture and the prior over the weights. Thus, we can relate any correlation function in function space and weight space with statistics given by the prior, for instance the two-point correlation function is

$$\int \mathcal{D}f \underbrace{\int dw P_0(w)\delta[f - z_w]}_{P_0[f]} f(x)f(x')$$

$$= \int dw P_0(w)z_w(x)z_w(x'), \quad (4)$$

where $\mathcal{D}f$ is the integration measure over function space. As noted by Cho and Saul [30], in the limit of $N \to \infty$ the right-hand side (RHS) of Eq. (4) equals the kernel of the NNGP associated with this DNN, $K(x, x')$. Moreover, for large but finite N, $P_0[f]$ will tend to a Gaussian with a $1/N$ leading correction

$$P_0[f] \propto e^{-\frac{1}{2}\int d\mu(x)d\mu(x')f(x)K^{-1}(x,x')f(x')} + O(1/N), \quad (5)$$

where $\mu(x)$ is the the probability measure from which the inputs x of the train and test sets are sampled: $d\mu(x) = P(x)dx$, and the $O(1/N)$ scaling of the finite-N correction will be explained in Sec. III. If we now plug the prior Eq. (5) in the expression for the posterior Eq. (3), take the loss to be the total square error [31] $\mathcal{L}[f] = \sum_{\alpha=1}^n [y_\alpha - f(x_\alpha)]^2$, and take $N \to \infty$ we have that the posterior $P[f]$ is that of a GP. Assuming ergodicity, one finds that training-time averaged output of the DNN is given by the posterior mean of a GP, with measurement noise [32] equal to $\sigma^2 = T/2$ and a kernel given by the NNGP of that DNN.

We refer to the above expressions for $P_0[f]$ and $P[f]$ describing the distribution of outputs of a DNN trained according to our protocol—the *NNSP correspondence*. Unlike the NTK correspondence, the kernel which appears here is different and no additional initialization dependent terms appear (as should be the case since we assumed ergodicity). Furthermore, given knowledge of $P_0[f]$ at finite N, one can predict the DNN's outputs at finite N. Henceforth, we refer to $P_0[f]$ as the prior distribution, as it is the prior distribution of a DNN with random weights drawn from $P_0(w)$.

C. Evidence supporting ergodicity

Our derivation relies on the ergodicity of the dynamics. Ergodicity is in general hard to prove rigorously in nonconvex settings, and thus we must revert to heuristic arguments. First, note that we are mainly interested in estimating the posterior mean of the outputs, thus we do not require full ergodicity but rather the much weaker condition of *ergodicity in the mean* (see Appendix F). The most robust evidence of ergodicity in the mean in function space is the high level of accuracy of our analytical expressions, Eq. (11), in predicting the numerical results.

Another indicator of ergodicity is a short autocorrelation time (ACT) of the dynamics. Short ACT does not logically imply ergodicity. However, the empirical ACT gives a lower bound on the true correlation time of the dynamics. In our framework, it is sufficient that the dynamics of the outputs z_w be ergodic, even if the dynamics of the weights converge much slower to an equilibrium distribution. Indeed, we have
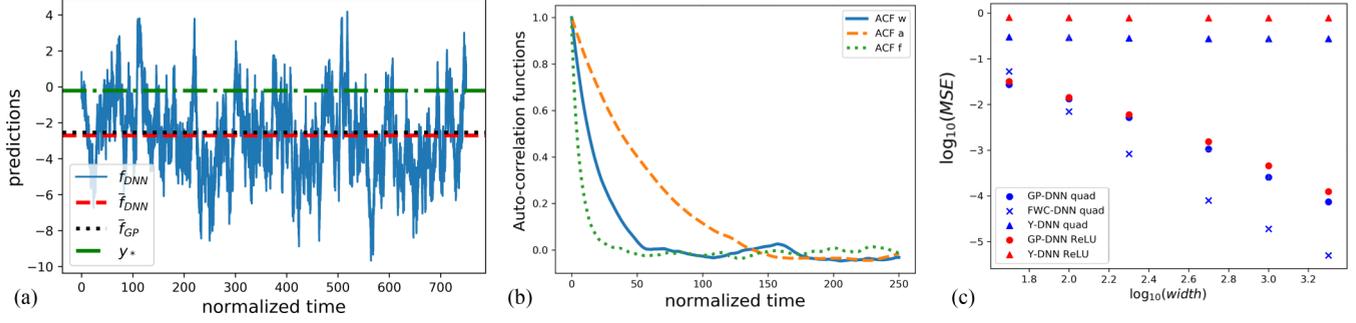
FIG. 2. Fully connected two-layer network trained on a regression task. (a) Network outputs on a test point $f_{\text{DNN}}(x_*, t)$ vs normalized time: the time-averaged DNN output $\bar{f}_{\text{DNN}}(x_*)$ (dashed line) is much closer to the GP prediction $\bar{f}_{\text{GP}}(x_*)$ (dotted line) than to the ground truth y_* (dashed-dotted line). (b) ACFs of the time series of the first- and second-layer weights, and of the outputs: the output converges to equilibrium faster than the weights. (c) Relative MSE between the network outputs and the labels y (triangles), GP predictions $\bar{f}_{\text{GP}}(x_*)$ Eq. (6) (dots), and FWC predictions Eq. (10) (x's), shown vs width for quadratic (blue) and ReLU (red) activations. For sufficiently large widths ($N \gtrsim 500$) the slope of the GP-DNN MSE approaches -2.0 and the FWC-DNN MSE is further improved by more than an order of magnitude.

found that the ACTs of the outputs are considerably smaller than those of the weights [see Fig. 2(b)].

Last, from the point of view of constraint satisfaction problems, optimizing the train loss can be seen as an attempt to find a solution to n constraints using far more variables (roughly MN^2 where M is the number of layers). One typically expects ergodic behavior when the ratio of the number of variables to the number of constraints becomes much larger than one [33], which is the case in our over-parameterized setting. In this regime, it has been shown that the loss landscape is characterized by connected manifolds of low loss points rather than isolated local minima [34], which further supports ergodicity.

D. Comparison between SGD and our training protocol

In this subsection we compare the training dynamics that arise as a result of more standard Stochastic Gradient Descent (SGD) algorithms and that of our training protocol involving full-batch gradient descent (GD), weight decay, and additive white Gaussian noise.

As there is no single standard training protocol for DNN training, for the sake of concreteness we will compare our protocol with SGD with zero momentum and finite weight decay. The primary difference between such SGD and our training protocol is the finite learning rate and source and character of the noise involved: in the former, the noise is a result of the random choice of mini-batch for each gradient step whereas in the latter the noise is externally injected at every step. SGD noise is *state-dependent*, namely, it depends on the current value of the DNN parameter vector. Additionally, mini-batch noise is anisotrpic with respect to the coordinates of the DNN weight space, and also correlated across training time, if we assume that each training example is sampled once per epoch (in line with best practice [35]). In contrast, the noise in our protocol is state-independent, isotropic and uncorrelated across training time. These properties ensure that the dynamics in Eq. (1) lead to a well-behaved and analytically tractable equilibrium distribution over weight space.

First we argue that for sufficiently small noise, large mini-batch size, and low learning rate, this qualitative difference would have only a small quantitative effect. Indeed, for small

noise we expect the initial trajectories under the two training protocols (SGD versus ours) to be very similar and dominated by the strong average gradient, until approaching the bottom of some basin of attraction. As aforementioned, in the over-parameterized regime, the landscape is "well-behaved": rather than isolated local minima we expect to find connected manifolds of low loss points [34]. Under these conditions, the difference between SGD and our protocol would be what is considered "wide minima" due to the anisotropic nature of the SGD noise. However, for small enough noise, all minima would be wide under both settings.

This prompts the question of whether going to small noise/small learning rate misses out on some generic performance boost. Here there is empirical evidence that moderate learning rates provide a consistent performance boost over vanishing learning rates even at vanishing noise [36–38]. Such finite learning rate effects are absent from our current analysis. While quantitatively important, here we take the viewpoint they are secondary in importance and focus, as various other authors do [2,39], on the Bayesian and GP picture of deep learning which emerges at low learning rate. Notwithstanding, as opposed to generic state-dependent noise, finite learning rates are more theoretically tractable. In particular, one can imagine incorporating these into our formalism by considering the finite learning-rate modifications to loss function studied in Ref. [40].

III. INFERENCE ON THE RESULTING NNSP

Having mapped the noise- and time-averaged outputs of a DNN to inference on the above NNSP, we turn to analyze the predictions of this NNSP in the case where N is large but finite, such that the NNSP is only weakly non-Gaussian [i.e., its deviation from a GP is $O(1/N)$]. Recall the standard GP regression results for the posterior mean $\bar{f}_{\text{GP}}(x_*)$ and variance $\Sigma_{\text{GP}}(x_*)$ on an unseen test point x_*, given a training set $\{(x_\alpha, y_\alpha)\}_{\alpha=1}^n \subset \mathbb{R}^d \times \mathbb{R}$ [3]

$$\bar{f}_{\text{GP}}(x_*) = \sum_{\alpha,\beta} y_\alpha \tilde{K}_{\alpha\beta}^{-1} K_\beta^*, \quad \Sigma_{\text{GP}}(x_*) = K^{**} - \sum_{\alpha,\beta} K_\alpha^* \tilde{K}_{\alpha\beta}^{-1} K_\beta^*,$$

(6)

where $\tilde{K}_{\alpha\beta} := K(x_\alpha, x_\beta) + \sigma^2\delta_{\alpha\beta}$; $K_\alpha^* := K(x_*, x_\alpha)$; $K^{**} := K(x_*, x_*)$. The main result of this section is a derivation of leading FWCs to the above results.

A. Edgeworth expansion and perturbation theory

Our first task is to find how $P[f]$ changes compared to the Gaussian ($N \to \infty$) scenario. As the data-dependent part $e^{-\mathcal{L}[f]/2\sigma^2}$ is independent of the DNN, this amounts to obtaining finite width corrections to the prior $P_0[f]$. One way to characterize this is to perform an *Edgeworth expansion* of $P_0[f]$ [41,42]; we give a recap of this topic in Appendix A. In essence, an Edgeworth expansion is a formal series that characterises some probability distribution in terms of its cumulants, and is most commonly used when the distribution at hand can be written as a Gaussian plus some small correction terms. This is especially conducive in our context since for all DNNs with the last layer being fully connected, all odd cumulants vanish and the $2r$th cumulant scales as $1/N^{r-1}$. Consequently, at large N we can characterize $P_0[f]$ up to $O(1/N^2)$ by its second and fourth cumulants, $K(x_1, x_2)$ and $U(x_1, x_2, x_3, x_4)$, respectively. Thus, the leading-order correction to $P_0[f]$ reads

$$P_0[f] \propto e^{-S_{\mathrm{GP}}[f]}\left(1 - \frac{1}{N}S_U[f]\right) + O(1/N^2), \quad (7)$$

where the GP action S_{GP} and the first FWC action S_U are given by

$$S_{\mathrm{GP}}[f] = \frac{1}{2}\int d\mu_{1:2}\, f_{x_1}K_{x_1,x_2}^{-1}f_{x_2},$$

$$S_U[f] = -\frac{1}{4!}\int d\mu_{1:4}\, U_{x_1,x_2,x_3,x_4}H_{x_1,x_2,x_3,x_4}[f]. \quad (8)$$

Here, H is the fourth functional Hermite polynomial (see Appendix A), U is the fourth-order functional cumulant of the NN output [we take $U \sim O(1)$ to emphasize the scaling with N in Eqs. (7) and (10)], which depends on the choice of the activation function ϕ,

$$U_{x_1,x_2,x_3,x_4} = \varsigma_a^4(\langle\phi_1\phi_2\phi_3\phi_4\rangle - \langle\phi_1\phi_2\rangle\langle\phi_3\phi_4\rangle)[3], \quad (9)$$

where $\phi_\alpha := \phi[z_i^{\ell-1}(x_\alpha)]$ and the preactivations are $z_i^\ell(x) = b_i^\ell + \sum_{j=1}^{N_\ell}W_{ij}^\ell\phi[z_j^{\ell-1}(x)]$. The bracket notation [3] indicates summing over the three distinct pairings of the integers $\{1, \ldots, 4\}$. Here we distinguished between the scaled and nonscaled weight variances: $\sigma_a^2 = \varsigma_a^2/N$, where a are the weights of the last layer. Our shorthand notation for the integration measure over inputs means, e.g., $d\mu_{1:4} := d\mu(x_1)\cdots d\mu(x_4)$. We show in Appendix B 3 that our results are invariant under a change of measure, thus we can keep it arbitrary at this point.

Using perturbation theory, in Appendix B we compute the leading FWC to the posterior mean $\bar{f}(x_*)$ and variance $\langle(\delta f(x_*))^2\rangle$ on a test point x_*,

$$\bar{f}(x_*) = \bar{f}_{\mathrm{GP}}(x_*) + N^{-1}\bar{f}_U(x_*) + O(N^{-2}),$$

$$\langle(\delta f(x_*))^2\rangle = \Sigma_{\mathrm{GP}}(x_*) + N^{-1}\Sigma_U(x_*) + O(N^{-2}), \quad (10)$$

where

$$\bar{f}_U(x_*) = \frac{1}{6}\tilde{U}_{\alpha_1\alpha_2\alpha_3}^*(\tilde{y}_{\alpha_1}\tilde{y}_{\alpha_2}\tilde{y}_{\alpha_3} - 3\tilde{K}_{\alpha_1\alpha_2}^{-1}\tilde{y}_{\alpha_3}),$$

$$\Sigma_U(x_*) = \frac{1}{2}\tilde{U}_{\alpha_1\alpha_2}^{**}(\tilde{y}_{\alpha_1}\tilde{y}_{\alpha_2} - \tilde{K}_{\alpha_1\alpha_2}^{-1}), \quad (11)$$

where all repeating indices are summed over the training set (i.e., range over $\{1, \ldots, n\}$), denoting: $\tilde{y}_\alpha := \tilde{K}_{\alpha\beta}^{-1}y_\beta$, and defining

$$\tilde{U}_{\alpha_1\alpha_2\alpha_3}^* := U_{\alpha_1\alpha_2\alpha_3}^* - U_{\alpha_1\alpha_2\alpha_3\alpha_4}\tilde{K}_{\alpha_4\beta}^{-1}K_\beta^*,$$

$$\tilde{U}_{\alpha_1\alpha_2}^{**} := U_{\alpha_1\alpha_2}^{**} - \left(U_{\alpha_1\alpha_2\alpha_3}^* + \tilde{U}_{\alpha_1\alpha_2\alpha_3}^*\right)\tilde{K}_{\alpha_3\beta}^{-1}K_\beta^*, \quad (12)$$

where asterisks denote evaluation at a test point x_*, e.g., $U_{\alpha_1\alpha_2\alpha_3}^* = U(x_*, x_{\alpha_1}, x_{\alpha_2}, x_{\alpha_3})$ and $U_{\alpha_1\alpha_2}^{**} = U(x_*, x_*, x_{\alpha_1}, x_{\alpha_2})$. Equations (11) and (12) are one of our key analytical results, which are qualitatively different from the corresponding GP expressions in Eq. (6). The correction to the predictive mean $\bar{f}_U(x_*)$ has a linear term in y, which can be viewed as a correction to the GP kernel, but also a cubic term in y, unlike $\bar{f}_{\mathrm{GP}}(x_*)$ which is purely linear. The correction to the predictive variance $\Sigma_U(x_*)$ has quadratic terms in y, unlike $\Sigma_{\mathrm{GP}}(x_*)$ which is y-independent. $\tilde{U}_{\alpha_1\alpha_2\alpha_3}^*$ has a clear interpretation in terms of GP regression: If we consider the indices $\alpha_1, \alpha_2, \alpha_3$ as fixed, then $U_{\alpha_1\alpha_2\alpha_3}^*$ can be thought of as the ground truth value of a target function (analogous to y_*), and the second term on the RHS $U_{\alpha_1\alpha_2\alpha_3\alpha_4}\tilde{K}_{\alpha_4\beta}^{-1}K_\beta^*$ is then the GP prediction of $U_{\alpha_1\alpha_2\alpha_3}^*$ with the kernel K, where α_4 runs on the training set [compare to $\bar{f}_{\mathrm{GP}}(x_*)$ in Eq. (6)]. Thus, $\tilde{U}_{\alpha_1\alpha_2\alpha_3}^*$ is the *discrepancy* in predicting $U_{\alpha_1\alpha_2\alpha_3\alpha_4}$ using a GP with kernel K. In Sec. III B we study the behavior of $\bar{f}_U(x_*)$ as a function of n.

The posterior variance $\Sigma(x) = \langle(\delta f(x))^2\rangle$ has a clear interpretation in our correspondence: it is a measure of how much we can decrease the test loss by ensembling (see also Refs. [8,43,44]). Our procedure for generating empirical network outputs involves time-averaging over the training dynamics after reaching equilibrium and also over different realizations of noise and initial conditions (see Appendix F). This allows for a reliable comparison with our FWC theory for the mean. In principle, one could use the network outputs at the end of training without this averaging, in which case there will be fluctuations that will scale with $\Sigma(x_\alpha)$. Following this, one finds that the expected MSE test loss after training saturates is $n_*^{-1}\sum_{\alpha=1}^{n_*}\{\langle[\bar{f}(x_\alpha) - y(x_\alpha)]^2\rangle + \Sigma(x_\alpha)\}$, where n_* is the size of the test set.

B. Finite-width corrections for small and large data sets

The expressions in Eqs. (6) and (11) for the GP prediction and the leading FWC are explicit but only up to a potentially large matrix inversion, $\tilde{K}^{-1}$, which is computationally costly and can accumulate numerical error. These matrices also have a random component related to the arbitrary choice of the particular n training points, which ruins whatever symmetries were present in the covariance function $K(x, x')$. An insightful tool, used in the context of GPs, which solves both these issues is the *equivalent kernel* (EK) [3,45]; see also Appendix J for a short review. In essence, the discrete sums over the training set appearing in Eq. (6) are replaced by integrals over all input space, which together with a spectral decomposition of the kernel function $K(x, x') = \sum_i \lambda_i\psi_i(x)\psi_i(x')$ yields the well known result

$$\bar{f}_{\mathrm{GP}}^{\mathrm{EK}}(x_*) = \int d\mu(x')\sum_i \frac{\lambda_i\psi_i(x_*)\psi_i(x')}{\lambda_i + \sigma^2/n}g(x'). \quad (13)$$

The EK approximates the GP predictions at large n, after averaging on all draws of (roughly) n training points representing the target function. Even if one is interested in a particular dataset, fluctuations due to the choice of data-set are often negligible at large n [4]. Here we develop an extension of Eq. (13) for the NNSPs we find at large but finite N. In particular, we find the leading nonlinear correction to the EK result, i.e., the "EK analog" of Eq. (11). To this end, we consider the average predictions of an NNSP trained on an ensemble of data sets of size n', corresponding to n' independent draws from a distribution $\mu(x)$ over all possible inputs x. Following the steps in Appendix K we find

$$
\bar{f}_U^{\text{EK}}(x_*) = \frac{1}{6}\tilde{\delta}_{x_* x_1} U_{x_1, x_2, x_3, x_4} \left\{ \frac{n^3}{\sigma^6} \tilde{\delta}_{x_2 x_2'} g(x_2') \tilde{\delta}_{x_3 x_3'} g(x_3') \tilde{\delta}_{x_4 x_4'} g(x_4') \right.
$$
$$
\left. - \frac{3n^2}{\sigma^4} \tilde{\delta}_{x_2, x_3} \tilde{\delta}_{x_4, x_4'} g(x_4') \right\}, \qquad (14)
$$

where an integral $\int d\mu(x)$ is implicit for every pair of repeated x coordinates. We introduced the *discrepancy operator* $\tilde{\delta}_{xx'}$ which acts on some function φ as $\int d\mu(x') \tilde{\delta}_{xx'} \varphi(x') := \tilde{\delta}_{xx'} \varphi(x') = \varphi(x) - \bar{f}_{\text{GP}}^{\text{EK}}(x)$. Essentially, Eq. (14) is derived from Eq. (11) by replacing each $\tilde{K}^{-1}$ by $(n/\sigma^2)\tilde{\delta}$ and noticing that in this regime $\tilde{U}_{x_2, x_3, x_4}^*$ in Eq. (12) becomes $\tilde{\delta}_{x_* x_1} U_{x_1, x_2, x_3, x_4}$. Interestingly, $\bar{f}_U^{\text{EK}}(x_*)$ is written explicitly in terms of meaningful quantities: $\tilde{\delta}_{xx'} g(x')$ and $\tilde{\delta}_{x_* x_1} U_{x_1, x_2, x_3, x_4}$.

Equations (13) and (14) are valid for any weakly non-Gaussian process, including ones related to CNNs (where N corresponds to the number of channels). It can also be systematically extended to smaller values of n by taking into account higher terms in $1/n$, as in Cohen *et al.* [4]. At $N \to \infty$, we obtain the standard EK result, Eq. (13). It is basically a high-pass linear filter which filters out features of g that have support on eigenfunctions ψ_i associated with eigenvalues λ_i that are small relative to σ^2/n. We stress that the ψ_i, λ_i's are independent of any particular size n dataset but rather are a property of the average dataset. In particular, no computationally costly data dependent matrix inversion is needed to evaluate Eq. (13).

Turning to our FWC result, Eq. (14), it depends on $g(x)$ only via the discrepancy operator $\tilde{\delta}_{xx'}$. Thus, these FWCs would be proportional to the error of the DNN, at $N \to \infty$. In particular, perfect performance at $N \to \infty$, implies no FWC. Second, the DNN's average predictions act as a linear transformation on the target function combined with a cubic nonlinearity. Third, for $g(x)$ having support only on some finite set of eigenfunctions ψ_i of K, $\tilde{\delta}_{xx'} g(x')$ would scale as σ^2/n at very large n. Thus, the above cubic term would lose its explicit dependence on n. The scaling with n of this second term is less obvious, but numerical results suggest that $\tilde{\delta}_{x_2 x_3}$ also scales as σ^2/n, so that the whole expression in the $\{\cdots\}$ has no scaling with n. In addition, some decreasing behavior with n is expected due to the $\tilde{\delta}_{x_* x_1} U_{x_1, x_2, x_3, x_4}$ factor which can be viewed as the discrepancy in predicting U_{x, x_2, x_3, x_4}, at fixed x_2, x_3, x_4, based on n random samples (x_α's) of $U_{x_\alpha, x_2, x_3, x_4}$. In Fig. 1 we illustrate this behavior at large n and also find that for small n the FWC is small but increasing with n, implying that at large N FWCs are only important at intermediate values of n.

IV. NUMERICAL EXPERIMENTS

In this section we numerically test our analytical results. We first demonstrate that in the limit $N \to \infty$ the outputs of a FCN trained in the regime of the NNSP correspondence converge to a GP with a known kernel, and that the MSE between them scales as $\sim 1/N^2$ which is the scaling of the leading FWC squared. Second, we show that introducing the leading FWC term $N^{-1} \bar{f}_U(x_*)$, Eq. (11), further reduces this MSE by more than an order of magnitude. Third, we study the performance gap between finite CNNs and their corresponding NNGPs on CIFAR-10.

A. Toy example: Fully connected networks on synthetic data

We trained a two-layer FCN $f(x) = \sum_{i=1}^{N} a_i \phi(w^{(i)} \cdot x)$ on a quadratic target $y(x) = x^{\mathsf{T}} A x$ where the x's are sampled with a uniform measure from the hyper-sphere $\mathbb{S}_{d-1}(\sqrt{d})$; see Appendix G 1 for more details. Our settings are such that there are not enough training points to fully learn the target: Fig. 2(a) shows that the time averaged outputs (after reaching equilibrium) $\bar{f}_{\text{DNN}}(x_*)$ is much closer to the GP prediction $\bar{f}_{\text{GP}}(x_*)$ than to the ground truth y_*. Otherwise, the convergence of the network output to the corresponding NNGP as N grows [shown in Fig. 2(c)] would be trivial, since all reasonable estimators would be close to the target and hence close to each other.

In Fig. 2(c) we plot in log-log scale (with base 10) the MSE [normalized by $(\bar{f}_{\text{DNN}})^2$] between the predictions of the network $\bar{f}_{\text{DNN}}$ and the corresponding GP and FWC predictions for quadratic and ReLU activations. We find that indeed for sufficiently large widths ($N \gtrsim 500$) the slope of the GP-DNN MSE approaches -2.0 (for both ReLU and quadratic), which is expected from our theory, since the leading FWC scales as $1/N$. For smaller widths, higher order terms (in $1/N$) in the Edgeworth series Eq. (7) come into play. For quadratic activation, we find that our FWC result further reduces the MSE by more than an order of magnitude relative to the GP theory. We recognize a regime where the GP and FWC MSEs intersect at $N \lesssim 100$, below which our FWC actually increases the MSE, which suggests a scale of how large N needs to be for our leading FWC theory to hold.

B. Performance gap between finite CNNs and their corresponding NNGPs

Several papers have shown that the performance on image classification tasks of SGD-trained finite CNNs can surpass that of the corresponding GPs, be it NTK [9] or NNGP [14]. More recently, Lee *et al.* [46] emphasized that this performance gap depends on the procedure used to collapse the spatial dimensions of image-shaped data before the final readout layer: flattening the image into a one-dimensional vector (CNN-VEC) or applying global average pooling to the spatial dimensions (CNN-GAP). It was observed that while infinite FCNs and CNN-VEC networks outperform their respective finite networks, infinite CNN-GAP networks under-perform their finite-width counterparts, i.e., there exists a finite optimal width.

One notable margin, of about 17% accuracy on CIFAR10, was shown in Novak *et al.* [14] for the case of CNN with
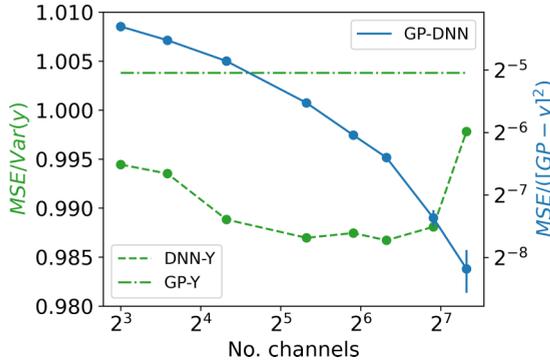
FIG. 3. DNN-GP MSE (blue) demonstrates convergence to a slope of -2.0, validating the theoretically expected scaling. DNN-ground truth (Y) MSE (green) shows finite CNN can outperform corresponding GP.

no pooling. It was further pointed out there, that the NNGPs associated with such CNNs, coincide with those of the corresponding locally connected networks (LCNs), namely, CNNs without weight sharing between spatial locations. Furthermore, the performance of SGD-trained LCNs was found to be on par with that of their NNGPs. We argue that our framework can account for this observation. The priors $P_0[f]$ of a LCN and CNN-VEC agree on their second cumulant (the covariance), which is the only one not vanishing as $N \to \infty$, but they need not agree on their higher order cumulants, which come into play at finite N. In Appendix I we show that U appearing in our leading FWC, already differentiates between CNNs and LCNs. Common practice strongly suggests that the prior over functions induced by CNNs is better suited than that of LCNs for classification of natural images. As a result we expect that the test loss of a finite-width CNN trained using our protocol will initially decrease with N but then increase beyond some *optimal width* N_{opt}, tending towards the loss of the corresponding GP as $N \to \infty$. This is in contrast to SGD behavior reported in some works where the CNN performance seems to saturate as a function of N, to some value better than the NNGP [14,47]. Notably those works used maximum over architecture scans, high learning rates, and early stopping, all of which are absent from our training protocol.

To test the above conjecture we trained, according to our protocol, a CNN with six convolutional layers and two fully connected layers on CIFAR10, and used CNN-VEC for the readout. We used MSE loss with a one-hot encoding into a 10 dimensional vector of the categorical label; further details and additional settings are given in Appendix G. Figure 3 demonstrates that, using our training protocol, a finite CNN can outperform its corresponding GP and approaches its GP as the number of channels increases. This phenomenon was observed in previous studies under realistic training settings [14], and here we show that it appears also under our training protocol. We note that a similar yet more pronounced trend in performance appears here also when one considers the averaged MSE loss rather the the MSE loss of the average outputs. Last, we comment that our analysis has focused on the simplest case with no pooling. It is an interesting open question if finite-width corrections improve performance for CNN architectures that include pooling.

V. CONCLUSION

In this work we presented a correspondence between finite-width DNNs trained using Langevin dynamics (i.e., using small learning rates, weight-decay and noisy gradients) and inference on a stochastic-process (the NNSP), which approaches the NNGP as $N \to \infty$. We derived finite width corrections, that improve upon the accuracy of the NNGP approximation for predicting the DNN outputs on unseen test points, as well as the expected fluctuations around these.

In the limit of a large number of training points $n \to \infty$, explicit expressions for the DNNs' outputs were given, involving no costly matrix inversions. In this regime, the FWC can be written in terms of the discrepancy of GP predictions, so that when GP has a small test error the FWC will be small, and vice versa. In the small n regime, the FWC is small but grows with n. Our formalism relates to the observation that finite CNNs (with no pooling layers) outperform their corresponding NNGPs on image classification tasks [14]. The weight-sharing property of finite CNNs is absent at the level of the NNGP but is reflected already in our leading FWCs. This constitutes one real-world example where the FWC is well suited to the structure of the data distribution, and thus improves performance relative to the corresponding GP.

There are several factors that make finite SGD-trained DNNs used in practice different from their GP counterparts, e.g., large learning rates, early stopping, etc. [46]. Importantly, our framework quantifies the specific contribution of finite-width effects to this difference, distilling it from the contribution of these other factors. In a future study, it would be very interesting to consider well-controlled toy models that can elucidate under what conditions on the architecture and data distribution does the FWC improve performance relative to GP.

ACKNOWLEDGMENTS

APPENDIX A: EDGEWORTH SERIES

In this Appendix we give a recap of the Edgeworth series expansion of some probability distribution, which is a way to characterize it in terms of its cumulants. We begin with scalar-valued RVs and then move on to discuss vector-valued RVs and finally distributions over function spaces, i.e., *stochastic processes*, which are the focus of this work.

1. Edgeworth expansion for a scalar random variable

Consider scalar valued continuous iid RVs $\{Z_i\}$ and assume WLOG $\langle Z_i \rangle = 0$, $\langle Z_i^2 \rangle = 1$, with higher cumulants κ_r^Z for $r \geqslant 3$. Now consider their normalized sum $Y_N = \frac{1}{\sqrt{N}} \sum_{i=1}^N Z_i$. Recall that cumulants are *additive*, i.e., if Z_1, Z_2 are independent RVs, then $\kappa_r(Z_1 + Z_2) = \kappa_r(Z_1) + \kappa_r(Z_2)$ and that the rth cumulant is *homogeneous* of degree r, i.e., if c is any constant, then $\kappa_r(cZ) = c^r \kappa_r(Z)$. Combining additivity and

homogeneity of cumulants we have a relation between the cumulants of Z and Y,

$$\kappa_{r \geqslant 2} := \kappa_{r \geqslant 2}^Y = \frac{N \kappa_r^Z}{(\sqrt{N})^r} = \frac{\kappa_r^Z}{N^{r/2-1}}. \qquad (A1)$$

Now, let $\varphi(y) := (2\pi)^{-1/2} e^{-y^2/2}$ be the PDF of the standard normal distribution. The *characteristic function* of Y is given by the Fourier transform of its PDF $P(y)$ and is expressed via its cumulants

$$\hat{P}(t) := \mathcal{F}[P(y)] = \exp\left(\sum_{r=1}^{\infty} \kappa_r \frac{(it)^r}{r!}\right)$$

$$= \exp\left(\sum_{r=3}^{\infty} \kappa_r \frac{(it)^r}{r!}\right) \hat{\varphi}(t), \qquad (A2)$$

where the last equality holds since we assumed $\kappa_1 = 0$, $\kappa_2 = 1$ and $\hat{\varphi}(t) = e^{-\frac{t^2}{2}}$. From the CLT, we know that $P(y) \to \varphi(y)$ as $N \to \infty$. Taking the inverse Fourier transform $\mathcal{F}^{-1}$ has the effect of mapping $it \mapsto -\partial_y$, thus

$$P(y) = \exp\left[\sum_{r=3}^{\infty} \kappa_r \frac{(-\partial_y)^r}{r!}\right] \varphi(y) = \varphi(y)\left[1 + \sum_{r=3}^{\infty} \frac{\kappa_r}{r!} H_r(y)\right], \qquad (A3)$$

where $H_r(y)$ is the rth *probabilist's Hermite polynomial*, defined by

$$H_r(y) = (-)^r e^{y^2/2} \frac{d^r}{dy^r} e^{-y^2/2}, \qquad (A4)$$

e.g., $H_4(y) = y^4 - 6y^2 + 3$.

2. Edgeworth expansion for a vector valued random variable

Consider now the analogous procedure for vector-valued RVs in $\mathbb{R}^n$ (see Ref. [41]). We perform an Edgeworth expansion around a centered multivariate Gaussian distribution with covariance matrix $\kappa^{i,j}$,

$$\varphi(\vec{y}) = \frac{1}{(2\pi)^{d/2} \det(\kappa^{i,j})} \exp\left(-\frac{1}{2}\kappa_{i,j} y^i y^j\right), \qquad (A5)$$

where $\kappa_{i,j}$ is the matrix inverse of $\kappa^{i,j}$ and Einstein summation is used. The rth-order cumulant becomes a tensor with r indices, e.g., the analog of κ_4 is $\kappa^{i,j,k,l}$. The Hermite polynomials are now multi-variate polynomials, so that the first one is $H_i = \kappa_{i,j} y^j$ and the fourth one is

$$H_{ijkl}(\vec{y}) = e^{\frac{1}{2}\kappa_{i',j'} y^{i'} y^{j'}} \partial_i \partial_j \partial_k \partial_l e^{-\frac{1}{2}\kappa_{i',j'} y^{i'} y^{j'}}$$

$$= H_i H_j H_k H_l - H_i H_j \kappa_{k,l}[6] + \kappa_{i,j} \kappa_{k,l}[3], \qquad (A6)$$

where the postscript bracket notation is simply a convenience to avoid listing explicitly all possible partitions of the indices, e.g., $\kappa_{i,j}\kappa_{k,l}[3] = \kappa_{i,j}\kappa_{k,l} + \kappa_{i,k}\kappa_{j,l} + \kappa_{i,l}\kappa_{j,k}$.

In our context we are interested in even distributions where all odd cumulants vanish, due to the symmetric statistics of the last-layer weights, so the Edgeworth expansion would read

$$P(\vec{y}) = \exp\left(\frac{\kappa^{i,j,k,l}}{4!} \partial_i \partial_j \partial_k \partial_l + \cdots\right) \varphi(\vec{y})$$

$$= \varphi(\vec{y})\left(1 + \frac{\kappa^{i,j,k,l}}{4!} H_{ijkl} + \cdots\right). \qquad (A7)$$

3. Edgeworth expansion for a function valued random variable

In this subsection we extend the result of the Edgeworth expansion for vector-valued RVs (i.e., distributions over finite-dimensional vector spaces) to function-valued RVs (i.e., distributions over function space or infinite-dimensional vector spaces). Loosely speaking, a function $f(x)$ can be thought of as an infinite dimensional vector with its argument playing the role of a continuous index. Thus, the cumulants become "functional tensors," i.e., multivariate functions of the input x.

Let us recall that in our context we are interested in the distribution of the outputs of some fully connected neural network. For simplicity we focus on a two-layer network, but the derivation generalizes straightforwardly to networks of any depth. We are interested in the finite N corrections to the prior distribution $P_0[f]$, i.e., the distribution of the DNN output $f(x) = \sum_{i=1}^{N} a_i \phi(w_i^\mathsf{T} x)$, with $a_i \sim \mathcal{N}(0, \frac{\varsigma_a^2}{N})$ and $w_i \sim \mathcal{N}(0, \frac{\varsigma_w^2}{d} I)$. Because a has zero mean and a variance that scales as $1/N$, all odd cumulants are zero and the $2r$th cumulant scales as $1/N^{r-1}$. This holds true for any DNN having a fully connected last layer with variance scaling as $1/N$. Thus, the leading FWC to the prior $P_0[f]$ is

$$P_0[f] = \frac{1}{Z} e^{-S_{\mathrm{GP}}[f]}\left[1 + \frac{1}{4!} \int d\mu(x_1) \cdots d\mu(x_4) U(x_1, x_2, x_3, x_4) H[f; x_1, x_2, x_3, x_4]\right] + O(1/N^2), \qquad (A8)$$

where $S_{\mathrm{GP}}[f]$ is as in the main text Eq. (8) and the fourth Hermite functional tensor is

$$H[f] = \int d\mu(x_1') \cdots d\mu(x_4') K^{-1}(x_1, x_1') \cdots K^{-1}(x_4, x_4') f(x_1') \cdots f(x_4')$$

$$- K^{-1}(x_\alpha, x_\beta) \int d\mu(x_\mu') d\mu(x_\nu') K^{-1}(x_\mu, x_\mu') K^{-1}(x_\nu, x_\nu') f(x_\mu') f(x_\nu')[6]$$

$$+ K^{-1}(x_\alpha, x_\beta) K^{-1}(x_\mu, x_\nu)[3], \qquad (A9)$$

where by the integers in $[\cdot]$ we mean all possible combinations of this form, e.g.,

$$K_{\alpha\beta}^{-1} K_{\mu\nu}^{-1} = K_{12}^{-1} K_{34}^{-1} + K_{13}^{-1} K_{24}^{-1} + K_{14}^{-1} K_{23}^{-1}. \qquad (A10)$$

The $H[f]$ appearing in Eq. (A9) is the functional analog of the multivariate Hermite polynomial Eq. (A6).

APPENDIX B: FIRST-ORDER CORRECTION TO POSTERIOR MEAN AND VARIANCE

1. Posterior mean

The posterior mean with the leading FWC action is given by

$$\langle f(x_*) \rangle = \frac{\int \mathcal{D}f e^{-S[f]} f(x_*)}{\int \mathcal{D}f e^{-S[f]}} + O(1/N^2), \qquad \text{(B1)}$$

where the action is

$$S[f] = S_{\text{GP}}[f] + S_{\text{Data}}[f] + S_U[f],$$

$$S_{\text{Data}}[f] = \frac{1}{2\sigma^2} \sum_{\alpha=1}^{n} (f(x_\alpha) - y_\alpha)^2, \qquad \text{(B2)}$$

and where the $O(1/N^2)$ implies that we only treat the first-order Taylor expansion of $S[f]$, and where $S_{\text{GP}}[f]$, $S_U[f]$ are as in the main text Eq. (8). The general strategy is to bring the path integral $\int \mathcal{D}f$ to the front, so that we will get just correlation functions w.r.t. the Gaussian theory (including the

data term $S_{\text{Data}}[f]$) $\langle \cdots \rangle_0$, namely, the well-known results [3] for $\bar{f}_{\text{GP}}(x_*) = \langle f(x_*) \rangle_0$ and $\Sigma_{\text{GP}}(x_*) = \langle (\delta f(x_*))^2 \rangle_0$, and then finally perform the integrals over input space. Expanding both the numerator and the denominator of Eq. (B1), the leading finite width correction for the posterior mean reads

$$\bar{f}_U(x_*) = \frac{1}{4!} \left(\int d\mu_{1:4} U(x_1, x_2, x_3, x_4) \langle f(x_*) H[f] \rangle_0 \right.$$
$$\left. - \langle f(x_*) \rangle_0 \int d\mu_{1:4} U(x_1, x_2, x_3, x_4) \langle H[f] \rangle_0 \right). \qquad \text{(B3)}$$

This, as standard in field theory, amounts to omitting all terms corresponding to bubble diagrams, namely, we keep only terms with a factor of $\langle f(x_*) f(x'_\alpha) \rangle_0$ and ignore terms with a factor of $\langle f(x_*) \rangle_0$, since these will cancel out. This is a standard result in perturbative field theory (see, e.g., Zee [48]).

We now write down the contributions of the quartic, quadratic and constant terms in $H[f]$:

(1) For the quartic term in $H[f]$, we have

$$\langle f(x_*) f(x'_1) f(x'_2) f(x'_3) f(x'_4) \rangle_0 - \langle f(x_*) \rangle_0 \langle f(x'_1) f(x'_2) f(x'_3) f(x'_4) \rangle_0$$
$$= \Sigma(x_*, x'_\alpha) \Sigma(x'_\beta, x'_\gamma) \bar{f}(x'_\delta)[12] + \Sigma(x_*, x'_\alpha) \bar{f}(x'_\beta) \bar{f}(x'_\gamma) \bar{f}(x'_\delta)[4]. \qquad \text{(B4)}$$

We dub these terms by $\bar{f} \Sigma \Sigma_*$ and $\bar{f} \bar{f} \bar{f} \Sigma_*$ to be referenced shortly. We mention here that they are the source of the linear and cubic terms in the target y appearing in Eq. (11) in the main text.

(2) For the quadratic term in $H[f]$, we have

$$\langle f(x_*) f(x'_\mu) f(x'_\nu) \rangle_0 - \langle f(x_*) \rangle_0 \langle f(x'_\mu) f(x'_\nu) \rangle_0$$
$$= \Sigma(x_*, x'_\mu) \bar{f}(x'_\nu)[2]. \qquad \text{(B5)}$$

We note in passing that these cancel out exactly together with similar but opposite sign terms/diagrams in the quartic contribution, which is a reflection of measure invariance. This is elaborated on in Appendix B 3.

(3) For the constant terms in $H[f]$, we will be left only with bubble diagram terms $\propto \int \mathcal{D}f \, f(x_*)$ which will cancel out in the leading order of $1/N$.

2. Posterior variance

The posterior variance is given by

$$\Sigma(x_*) = \langle f(x_*) f(x_*) \rangle - \bar{f}^2$$
$$= \langle f(x_*) f(x_*) \rangle_0 + \langle f(x_*) f(x_*) \rangle_U$$
$$\quad - \bar{f}_{\text{GP}}^2 - 2\bar{f}_{\text{GP}} \bar{f}_U + O(1/N^2)$$
$$= \Sigma_{\text{GP}}(x_*) + \langle f(x_*) f(x_*) \rangle_U - 2\bar{f}_{\text{GP}} \bar{f}_U + O(1/N^2). \qquad \text{(B6)}$$

Following similar steps as for the posterior mean, the leading finite width correction for the posterior second moment at

x_* reads

$$\langle f(x_*) f(x_*) \rangle_U$$
$$= \frac{1}{4!} \left(\int d\mu_{1:4} U(x_1, x_2, x_3, x_4) \langle f(x_*) f(x_*) H[f] \rangle_0 \right.$$
$$\left. - \langle f(x_*) f(x_*) \rangle_0 \int d\mu_{1:4} U(x_1, x_2, x_3, x_4) \langle H[f] \rangle_0 \right). \qquad \text{(B7)}$$

As for the posterior mean, the constant terms in $H[f]$ cancel out and the contributions of the quartic and quadratic terms are

$$\text{quartic terms} = \Sigma_{*\alpha} \Sigma_{*\beta} \bar{f}_\gamma \bar{f}_\delta [12] + \Sigma_{*\alpha} \Sigma_{*\beta} \Sigma_{\gamma\delta}[12] \quad \text{(B8)}$$

and

$$\text{quadratic terms} = \Sigma_{*\mu} \Sigma_{*\nu}[2]. \qquad \text{(B9)}$$

The quartic terms exactly cancel out with the $-2\bar{f}_{\text{GP}} \bar{f}_U$ terms and we are left with only quadratic contributions, as in the main text.

3. Measure invariance of the result

The expressions derived above may seem formidable, since they contain many terms and involve integrals over input space which seemingly depend on the measure $\mu(x)$. Here we show how they may in fact be simplified to the compact expressions in the main text Eq. (11) which involve only discrete sums over the training set and no integrals, and are thus manifestly measure-invariant.

For simplicity, we show here the derivation for the FWC of the mean $\bar{f}_U(x_*)$, and a similar derivation can be done for $\Sigma_U(x_*)$. In the following, we carry out the x integrals, by

plugging in the expressions from Eq. (6) and coupling them to U. As in the main text, we use the Einstein summation notation, i.e., repeated indices are summed over the training set. The contribution of the quadratic terms is

$$A_{\alpha_1,*} \tilde{K}^{-1}_{\alpha_1\beta_1} y_{\beta_1} - A_{\alpha_1\alpha_2} \tilde{K}^{-1}_{\alpha_1\beta_1} \tilde{K}^{-1}_{\alpha_2\beta_2} y_{\beta_1} K_{\beta_2,*}, \quad \text{(B10)}$$

where we defined

$$A(x_3, x_4) := \iint d\mu(x_1) d\mu(x_2) U(x_1, x_2, x_3, x_4) K^{-1}(x_1, x_2). \quad \text{(B11)}$$

Fortunately, this seemingly measure-dependent expression will cancel out with one of the terms coming from the $\tilde{f}\Sigma\Sigma_*$ contribution of the quartic terms in $H[f]$. This is not a

coincidence and is a general feature of the Hermite polynomials appearing in the Edgeworth series, thus for any order in $1/N$ in the Edgeworth series we will always be left only with measure invariant terms. Collecting all terms that survive we have

$$\frac{1}{4!}\Big\{4\tilde{U}^*_{\alpha_1\alpha_2\alpha_3} \tilde{K}^{-1}_{\alpha_1\beta_1} \tilde{K}^{-1}_{\alpha_2\beta_2} \tilde{K}^{-1}_{\alpha_3\beta_3} y_{\beta_1} y_{\beta_2} y_{\beta_3}$$
$$- 12\tilde{U}^*_{\alpha_1\alpha_2\alpha_3} \tilde{K}^{-1}_{\alpha_2\beta_2} \tilde{K}^{-1}_{\alpha_1\beta_1} y_{\beta_1}\Big\}, \quad \text{(B12)}$$

where we defined

$$\tilde{U}^*_{\alpha_1\alpha_2\alpha_3} := U^*_{\alpha_1\alpha_2\alpha_3} - U_{\alpha_1\alpha_2\alpha_3\alpha_4} \tilde{K}^{-1}_{\alpha_4\beta_4} K^*_{\beta_4}. \quad \text{(B13)}$$

This is a more explicit form of the result reported in the main text, Eq. (11).

APPENDIX C: FINITE WIDTH CORRECTIONS FOR MORE THAN ONE HIDDEN LAYER

For simplicity, consider a fully connected network with two hidden layers both of width N, and no biases, thus the preactivations $h(x)$ and output $z(x)$ are given by

$$h(x) = \frac{\sigma_{w_2}}{\sqrt{N}} W^{(2)} \phi\left(\frac{\sigma_{w_1}}{\sqrt{d}} W^{(1)} x\right), \quad z(x) = \frac{\sigma_a}{\sqrt{N}} a^{\mathsf{T}} \phi[h^{(2)}(x)]. \quad \text{(C1)}$$

We want to find the second and fourth cumulants of $z(x)$. Recall that we found that the leading-order Edgeworth expansion for the functional distribution of h is

$$P_{K,U}[h] \propto e^{-\frac{1}{2}h(x_1')K^{-1}(x_1',x_2')h(x_2')}\left(1 + \frac{1}{N}U(x_1', x_2', x_3', x_4')H[h; x_1', x_2', x_3', x_4']\right), \quad \text{(C2)}$$

where $K^{-1}(x_1', x_2')$ and $U(x_1', x_2', x_3', x_4')$ are known from the previous layer. So we are looking for two maps:

$$\mathcal{K}_\phi(K, U)(x, x') = \langle\phi[h(x)]\phi[h(x')]\rangle_{P_{K,U}[h]}, \quad \mathcal{U}_\phi(K, U)(x_1, x_2, x_3, x_4) = \langle\phi[h(x_1)]\phi[h(x_2)]\phi[h(x_3)]\phi[h(x_4)]\rangle_{P_{K,U}[h]}, \quad \text{(C3)}$$

so that the mapping between the first two cumulants K and U of two consequent layers is (assuming no biases)

$$\frac{K^{(\ell+1)}(x, x')}{\sigma^2_{w^{(\ell+1)}}} = \mathcal{K}_\phi(K^{(\ell)}, U^{(\ell)})(x, x'),$$

$$\frac{U^{(\ell+1)}(x_1, x_2, x_3, x_4)}{\sigma^4_{w^{(\ell+1)}}} = \mathcal{U}_\phi(K^{(\ell)}, U^{(\ell)})(x_1, x_2, x_3, x_4) - \mathcal{K}_\phi(K^{(\ell)}, U^{(\ell)})(x_{\alpha_1}, x_{\alpha_2})\mathcal{K}_\phi(K^{(\ell)}, U^{(\ell)})(x_{\alpha_3}, x_{\alpha_4})[3], \quad \text{(C4)}$$

where the starting point is the first layer ($N^{(0)} \equiv d$),

$$K^{(1)}(x, x') = \frac{\sigma^2_{w^{(1)}}}{N^{(0)}} x \cdot x' \quad U^{(1)}(x_1, x_2, x_3, x_4) = 0. \quad \text{(C5)}$$

The important point to note, is that these functional integrals can be reduced to ordinary finite dimensional integrals. For example, for the second layer, denote

$$\mathbf{h} := \begin{pmatrix} h_1 \\ h_2 \end{pmatrix} \quad \mathbf{K^{(1)}} = \begin{pmatrix} K^{(1)}(x_1, x_1) & K^{(1)}(x_1, x_2) \\ K^{(1)}(x_1, x_2) & K^{(1)}(x_2, x_2) \end{pmatrix}, \quad \text{(C6)}$$

we find for $K^{(2)}$

$$\frac{K^{(2)}(x_1, x_2)}{\sigma^2_{w^{(2)}}} = \int d\mathbf{h}\, e^{-\frac{1}{2}\mathbf{h}^{\mathsf{T}}\mathbf{K}^{-1}_{(1)}\mathbf{h}} \phi(h_1)\phi(h_2), \quad \text{(C7)}$$

and for $U^{(2)}$ we denote

$$\mathbf{h} := \begin{pmatrix} h_1 \\ h_2 \\ h_3 \\ h_4 \end{pmatrix} \quad \mathbf{K^{(1)}} = \begin{pmatrix} K^{(1)}(x_1, x_1) & K^{(1)}(x_1, x_2) & K^{(1)}(x_1, x_3) & K^{(1)}(x_1, x_4) \\ K^{(1)}(x_1, x_2) & K^{(1)}(x_2, x_2) & K^{(1)}(x_2, x_3) & K^{(1)}(x_2, x_4) \\ K^{(1)}(x_1, x_3) & K^{(1)}(x_2, x_3) & K^{(1)}(x_3, x_3) & K^{(1)}(x_3, x_4) \\ K^{(1)}(x_1, x_4) & K^{(1)}(x_2, x_4) & K^{(1)}(x_3, x_4) & K^{(1)}(x_4, x_4) \end{pmatrix}, \quad \text{(C8)}$$

so that

$$\mathcal{U}_\phi(K^{(1)}, U^{(1)})(x_1, x_2, x_3, x_4) = \int d\mathbf{h}\, e^{-\frac{1}{2}\mathbf{h}^{\mathsf{T}}\mathbf{K}^{-1}_{(1)}\mathbf{h}} \phi(h_1)\phi(h_2)\phi(h_3)\phi(h_4). \quad \text{(C9)}$$

This iterative process can be repeated for an arbitrary number of layers.

APPENDIX D: FOURTH CUMULANT FOR THRESHOLD POWER-LAW ACTIVATION FUNCTIONS

1. Fourth cumulant for ReLU activation function

The U's appearing in our FWC results can be derived for several activations functions, and in our numerical experiments we use a quadratic activation $\phi(z) = z^2$ and ReLU. Here we give the result for ReLU, which is similar for any other threshold power law activation (see derivation in Appendix D 2), and give the result for quadratic activation in Appendix E. For simplicity, in this section we focus on the case of a two-layer FCN with no biases, input dimension d and N neurons in the hidden layer, such that $\phi_\alpha^i := \phi(w^{(i)} \cdot x_\alpha)$ is the activation at the ith hidden unit with input x_α sampled with a uniform measure from $\mathbb{S}_{d-1}(\sqrt{d})$, where $w^{(i)}$ is a vector of weights of the first layer. This can be generalized to the more realistic settings of deeper nets and un-normalized inputs, where in the former the linear kernel L is replaced by the kernel of the layer preceding the output, and the latter amounts to introducing some scaling factors.

For $\phi = \text{ReLU}$, Cho and Saul [30] give a closed form expression for the kernel which corresponds to the GP. Here we find U corresponding to the leading FWC by first finding the fourth moment of the hidden layer $\langle \phi_1 \phi_2 \phi_3 \phi_4 \rangle$ [see Eq. (9)], taking for simplicity $\varsigma_w^2 = 1$,

$$\langle \phi_1 \phi_2 \phi_3 \phi_4 \rangle = \frac{\sqrt{\det(L^{-1})}}{(2\pi)^2} \int_0^\infty d\mathbf{z} e^{-\frac{1}{2}\mathbf{z}^\mathsf{T} L^{-1} \mathbf{z}} z_1 z_2 z_3 z_4, \tag{D1}$$

where L^{-1} above corresponds to the matrix inverse of the 4×4 matrix with elements $L_{\alpha\beta} = (x_\alpha \cdot x_\beta)/d$ which is the kernel of the previous layer (the linear kernel in the two-layer case) evaluated on two random points. In Appendix D 2 we follow the derivation in Moran [49], which yields (with a slight modification noted therein) the following series in the off-diagonal elements of the matrix L:

$$\langle \phi_1 \phi_2 \phi_3 \phi_4 \rangle = \sum_{\ell,m,n,p,q,r=0}^\infty A_{\ell mnpqr} L_{12}^\ell L_{13}^m L_{14}^n L_{23}^p L_{24}^q L_{34}^r, \tag{D2}$$

where the coefficients $A_{\ell mnpqr}$ are

$$\frac{(-)^{\ell+m+n+p+q+r} G_{\ell+m+n} G_{\ell+p+q} G_{m+p+r} G_{n+q+r}}{\ell! m! n! p! q! r!}. \tag{D3}$$

For ReLU activation, these G's read

$$G_s^{\text{ReLU}} = \begin{cases} \frac{1}{\sqrt{2\pi}} & s = 0, \\ \frac{-i}{2} & s = 1, \\ 0 & s \geqslant 3 \text{ and odd}, \\ \frac{(-)^k (2k)!}{\sqrt{2\pi} 2^k k!} & s = 2k+2 \quad k = 0, 1, 2, ..., \end{cases} \tag{D4}$$

and similar expressions can be derived for other threshold power-law activations of the form $\phi(z) = \Theta(z) z^\nu$. The series Eq. (D2) is expected to converge for sufficiently large input dimension d since the overlap between random normalized inputs scales as $\mathcal{O}(1/\sqrt{d})$ and consequently $L(x, x') \sim O(1/\sqrt{d})$ for two random points from the data sets. However, when we sum over $U_{\alpha_1...\alpha_4}$ we also have terms with repeating indices and so $L_{\alpha\beta}$'s are equal to 1. The above Taylor expansion diverges whenever the 4×4 matrix $L_{\alpha\beta} - \delta_{\alpha\beta}$ has eigenvalues larger than 1. Notably this divergence does not reflect a true divergence of U, but rather the failure of representing it using the above expansion. Therefore, at large n, one can opt to neglect elements of U with repeating indices, since there are much fewer of these. Alternatively this can be dealt with by a re-parameterization of the z's leading to a similar but slightly more involved Taylor series.

2. Derivation of the previous subsection

In this section we derive the expression for the fourth moment $\langle f_1 f_2 f_3 f_4 \rangle$ of a two-layer fully connected network with threshold-power law activations with exponent ν: $\phi(z) = \Theta(z) z^\nu$; $\nu = 0$ corresponds to a step function, $\nu = 1$ corresponds to ReLU, $\nu = 2$ corresponds to ReQU (rectified quadratic unit) and so forth.

When the inputs are normalized to lie on the hypersphere, the matrix L is

$$L = \begin{pmatrix} 1 & L_{12} & L_{13} & L_{14} \\ L_{12} & 1 & L_{23} & L_{24} \\ L_{13} & L_{23} & 1 & L_{34} \\ L_{14} & L_{24} & L_{34} & 1 \end{pmatrix}, \tag{D5}$$

where the off diagonal elements here have $L_{\alpha\beta} = O(1/\sqrt{d})$. We follow the derivation in Moran [49], which computes the probability mass of the positive orthant for a quadrivariate Gaussian distribution with covariance matrix L:

$$P_+ = \frac{\sqrt{\det(L^{-1})}}{(2\pi)^2} \int_0^\infty d\mathbf{z} e^{-\frac{1}{2}\mathbf{z}^{\mathsf{T}} L^{-1}\mathbf{z}}. \tag{D6}$$

The characteristic function (Fourier transform) of this distribution is

$$
\begin{aligned}
\varphi(t_1, t_2, t_3, t_4) &= \exp\left(-\frac{1}{2}\mathbf{t}^{\mathsf{T}} L\mathbf{t}\right) \\
&= \exp\left(-\frac{1}{2}\sum_{\alpha=1}^4 t_\alpha^2\right) \exp\left(-\sum_{\alpha<\beta} L_{\alpha\beta} t_\alpha t_\beta\right) \\
&= \exp\left(-\frac{1}{2}\sum_{\alpha=1}^4 t_\alpha^2\right) \sum_{\ell,m,n,p,q,r=0}^\infty \frac{(-)^{\ell+m+n+p+q+r} L_{12}^\ell L_{13}^m L_{14}^n L_{23}^p L_{24}^q L_{34}^r}{\ell!m!n!p!q!r!} t_1^{\ell+m+n} t_2^{\ell+p+q} t_3^{m+p+r} t_4^{n+q+r}.
\end{aligned}
\tag{D7}
$$

Performing an inverse Fourier transform, we may now write the positive orthant probability as

$$
\begin{aligned}
P_+ &= \frac{1}{(2\pi)^4} \int_{\mathbb{R}_+^4} d\mathbf{z} \int_{\mathbb{R}^4} d\mathbf{t}\; \varphi(t_1, t_2, t_3, t_4) e^{-i\sum_{\alpha=1}^4 z_\alpha t_\alpha} \\
&= \sum_{\ell,m,n,p,q,r=0}^\infty \frac{(-)^{\ell+m+n+p+q+r} L_{12}^\ell L_{13}^m L_{14}^n L_{23}^p L_{24}^q L_{34}^r}{\ell!m!n!p!q!r!} \cdots \frac{1}{(2\pi)^4} \int_{\mathbb{R}_+^4} d\mathbf{z} \int_{\mathbb{R}^4} d\mathbf{t}\; e^{\sum_{\alpha=1}^4 \left(-\frac{1}{2}t_\alpha^2 - iz_\alpha t_\alpha\right)} t_1^{\ell+m+n} t_2^{\ell+p+q} t_3^{m+p+r} t_4^{n+q+r} \\
&= \sum_{\ell,m,n,p,q,r=0}^\infty A_{\ell mnpqr} L_{12}^\ell L_{13}^m L_{14}^n L_{23}^p L_{24}^q L_{34}^r,
\end{aligned}
\tag{D8}
$$

where the coefficients $A_{\ell mnpqr}$ are

$$A_{\ell mnpqr} = \frac{(-)^{\ell+m+n+p+q+r} G_{\ell+m+n} G_{\ell+p+q} G_{m+p+r} G_{n+q+r}}{\ell!m!n!p!q!r!}, \tag{D9}$$

and the one dimensional integral is

$$G_s^{(\nu=0)} = \frac{1}{2\pi} \int_0^\infty dz \int_{-\infty}^\infty t^s \exp\left(-\frac{1}{2}t^2 - itz\right) dt. \tag{D10}$$

We can evaluate the integral over t to get

$$G_s^{(\nu=0)} = \frac{1}{(-i)^s(2\pi)^{1/2}} \int_0^\infty \left(\frac{d}{dz}\right)^s e^{-z^2/2} dz, \tag{D11}$$

and performing the integral over z yields

$$G_s^{(\nu=0)} = \begin{cases} \frac{1}{2} & s = 0, \\ 0 & s \text{ even and } s \geqslant 2, \\ \frac{(2k)!}{i(2\pi)^{1/2} 2^k k!} & s = 2k+1 \quad k = 0, 1, 2, \dots. \end{cases} \tag{D12}$$

We can now obtain the result for any integer ν by inserting z^ν inside the z integral:

$$G_s^{(\nu)} = \frac{1}{2\pi} \int_0^\infty dz\, z^\nu \int_{-\infty}^\infty t^s \exp\left(-\frac{1}{2}t^2 - itz\right) dt = \frac{1}{(-i)^s(2\pi)^{1/2}} \int_0^\infty z^\nu \left(\frac{d}{dz}\right)^s e^{-z^2/2} dz. \tag{D13}$$

Using integration by parts we arrive at the result Eq. (D4) reported in the main text

$$G_s^{\text{ReLU}} = G_s^{(\nu=1)} = \begin{cases} \frac{1}{\sqrt{2\pi}} & s = 0, \\ \frac{-i}{2} & s = 1, \\ 0 & s \geqslant 3 \text{ and odd}, \\ \frac{(-)^k (2k)!}{\sqrt{2\pi} 2^k k!} & s = 2k+2 \quad k = 0, 1, 2, \dots. \end{cases} \tag{D14}$$

Similar expressions can be derived for other threshold power-law activations of the form $\phi(z) = \Theta(z)z^\nu$ for arbitrary integer ν. In a more realistic setting, the inputs x may not be perfectly normalized, in which case the diagonal elements of L are not

unity. It amounts to introducing a scaling factor for each of the four z's and makes the expressions a little less neat but poses no real obstacle.

APPENDIX E: FOURTH CUMULANT FOR QUADRATIC ACTIVATION FUNCTION

For a two-layer network, we may write U, the fourth cumulant of the output $f(x) = \sum_{i=1}^{N} a_i \phi(w_i^{\mathsf{T}} x)$, with $a_i \sim \mathcal{N}(0, \varsigma_a^2/N)$ and $w_i \sim \mathcal{N}(\mathbf{0}, (\varsigma_w^2/d)I)$ for a general activation function ϕ as

$$U_{\alpha_1,\alpha_2,\alpha_3,\alpha_4} = \frac{\varsigma_a^4}{N} \big(V_{(\alpha_1,\alpha_2),(\alpha_3,\alpha_4)} + V_{(\alpha_1,\alpha_3),(\alpha_2,\alpha_4)} + V_{(\alpha_1,\alpha_4),(\alpha_2,\alpha_3)} \big), \tag{E1}$$

with

$$V_{(\alpha_1,\alpha_2),(\alpha_3,\alpha_4)} = \langle \phi^{\alpha_1}\phi^{\alpha_2}\phi^{\alpha_3}\phi^{\alpha_4} \rangle_w - \langle \phi^{\alpha_1}\phi^{\alpha_2} \rangle_w \langle \phi^{\alpha_3}\phi^{\alpha_4} \rangle_w. \tag{E2}$$

For the case of a quadratic activation function $\phi(z) = z^2$ the V's read

$$\begin{aligned}
V_{(\alpha_1,\alpha_2),(\alpha_3,\alpha_4)} = {} & 2\{L_{11}L_{33}(L_{24})^2 + L_{11}L_{44}(L_{23})^2 + L_{22}L_{33}(L_{14})^2 + L_{22}L_{44}(L_{13})^2\} + \ldots 4\{(L_{13})^2(L_{24})^2 + (L_{14})^2(L_{23})^2\} \\
& + 8(L_{11}L_{23}L_{34}L_{24} + L_{22}L_{34}L_{14}L_{13} + L_{33}L_{12}L_{14}L_{24} + L_{44}L_{12}L_{13}L_{23}) + \ldots 16(L_{12}L_{13}L_{24}L_{34} + L_{12}L_{14}L_{23}L_{34} \\
& + L_{13}L_{14}L_{23}L_{24}),
\end{aligned} \tag{E3}$$

where the linear kernel from the first layer is $L(x, x') = \frac{\varsigma_w^2}{d} x \cdot x'$. Notice that we distinguish between the scaled and nonscaled variances:

$$\sigma_a^2 = \frac{\varsigma_a^2}{N}; \quad \sigma_w^2 = \frac{\varsigma_w^2}{d}. \tag{E4}$$

These formulas were used when comparing the outputs of the empirical two-layer network with our FWC theory Eq. (11). One can generalize them straightforwardly to a network with M layers by recursively computing $K^{(M-1)}$ the kernel in the $(M-1)$th layer (see, e.g., Cho and Saul [30]), and replacing L with $K^{(M-1)}$.

APPENDIX F: AUTOCORRELATION TIME AND ERGODICITY

As mentioned in the main text, the network outputs $\bar{f}_{\text{DNN}}(x_*)$ are a result of averaging across many realizations (seeds) of initial conditions and the noisy training dynamics, and across time (epochs) after the training loss levels off. Our NNSP correspondence relies on the fact that our stochastic training dynamics are ergodic, namely, that averages across time equal ensemble averages. Actually, for our purposes it suffices that the dynamics are *ergodic in the mean*, namely, that the time-average estimate of the mean obtained from a single sample realization of the process converges in both the mean and in the mean-square sense to the ensemble mean:

$$\lim_{\bar{T}\to\infty} \mathbb{E}[\langle f^{\text{DNN}}(x_*; t)\rangle_{\bar{T}} - \mu(x_*)] = 0,$$
$$\lim_{\bar{T}\to\infty} \mathbb{E}[(\langle f^{\text{DNN}}(x_*; t)\rangle_{\bar{T}} - \mu(x_*))^2] = 0, \tag{F1}$$

where $\mu(x_*)$ is the ensemble mean on the test point x_* and the time-average estimate of the mean over a time window $\bar{T}$ is

$$\langle f^{\text{DNN}}(x_*; t)\rangle_{\bar{T}} := \frac{1}{\bar{T}} \int_0^{\bar{T}} f^{\text{DNN}}(x_*; t)\,dt \approx \frac{1}{\bar{T}} \sum_{t_j=0}^{t_j=\bar{T}} f^{\text{DNN}}(x_*; t_j). \tag{F2}$$

This is hard to prove rigorously but we can do a numerical consistency check using the following procedure: Consider the time series of the network output on the test point x_* for the ith realization as a row vector and stack these row vectors for all different realizations into a matrix F, such that $F_{ij} = f_i^{\text{DNN}}(x_*; t_j)$. (1) Divide the time series data in the matrix F into nonoverlapping submatrices, each of dimension

$n_{\text{seeds}} \times n_{\text{epochs}}$. (2) For each of these submatrices, find $\hat{f}(x_*)$, i.e., the empirical dynamical average across that time window and across the chosen seeds; (3) Find the empirical variance $\sigma_{\text{emp}}^2(x_*)$ across these $\hat{f}(x_*)$; (4) Repeat (1)–(3) for other combinations of $n_{\text{epochs}}, n_{\text{seeds}}$. If ergodicity holds, then we should expect to see the following relation:

$$\sigma_{\text{emp}}^2(x_*) = \sigma_m^2 \frac{\tau}{n_{\text{epochs}} n_{\text{seeds}}}, \tag{F3}$$

where τ is the autocorrelation time of the outputs and σ_m^2 is the macroscopic variance. The results of this procedure are shown in Fig. 4, where we plot on a log-log scale the empirical variance σ_{emp}^2 versus the number of epochs n_{epochs} used for time averaging in each set (and using all 500 seeds in this case). Performing a linear fit on the average across test points (black x's in the figure) yields a slope of approximately -1, which is strong evidence for ergodic dynamics.

APPENDIX G: NUMERICAL EXPERIMENT DETAILS

1. FCN experiment details

We trained a two-layer FCN on a quadratic target $y(x) = x^{\mathsf{T}} A x$ where the x's are sampled with a uniform measure from the hyper-sphere $\mathbb{S}_{d-1}(\sqrt{d})$, with $d = 16$ and the matrix elements are sampled as $A_{ij} \sim \mathcal{N}(0, 1)$ and fixed for all x's. For both activation functions, we used a training noise level of $\sigma^2 = 0.2$, training set of size $n = 110$ and a weight decay of the first layer $\gamma_w = 0.05$. Notice that for any activation ϕ, K scales linearly with $\varsigma_a^2 = \sigma_a^2 N = (T/\gamma_a) \cdot N$, thus to keep K constant as we vary N we need to scale the weight decay of the last layer as $\gamma_a \sim O(N)$. This is done to keep the prior
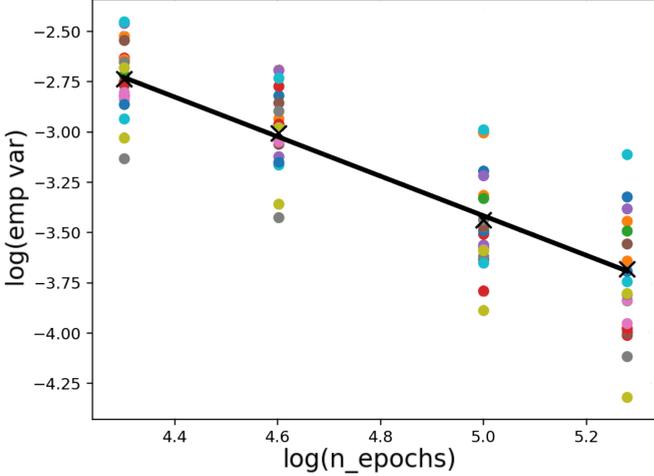
FIG. 4. Ergodicity check. Empirical variance $\sigma_{\text{emp}}^2(x_*)$ vs the number of epochs used for time averaging on a (base 10) log-log scale, with $\eta = 0.003$ and $N = 200$. The colored circles represent different test points x_* and the black x's are averages across these.

distribution in accord with the typical values of the target as N varies, so that the comparison is fair.

We ran each experiment for 2×10^6 epochs, which includes the time it takes for the training loss to level off, which is usually on the order of 10^4 epochs. In the main text we showed GP and FWC results for a learning rate of $\eta = 0.001$. Here we report in Fig. 5 the results using $\eta \in \{0.003, 0.001, 0.0005\}$. For a learning rate of $\eta = 0.003$ and width $N \geqslant 1000$ the dynamics become unstable and strongly oscillate, thus the general trend is broken, as seen in the blue markers in Fig. 5.
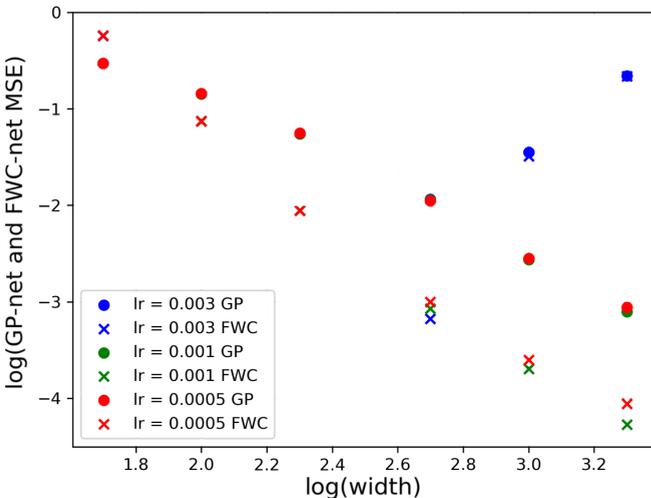
FIG. 5. Regression task with fully connected network: (un-normalized) MSE vs width on log-log scale (base 10) for quadratic activation and different leaning rates. The learning rates $\eta = 0.001$, 0.0005 converge to very similar values (recall this is a log scale), demonstrating that the learning rate is sufficiently small so that the discrete-time dynamics is a good approximation of the continuous-time dynamics. For a learning rate of $\eta = 0.003$ (blue) and width $N \geqslant 1000$ the dynamics become unstable, thus the general trend is broken, so one cannot take η to be too large.

The dynamics with the smaller learning rates are stable, and we see that there is a convergence to very similar values up to an expected statistical error.

2. CNN experiment details and additional settings

The CNN experiment reported in the main text was carried as follows.

Dataset. In the main text Fig. 3 we used a random sample of 10 train-points and 2000 test points from the CIFAR10 dataset, and in Appendix H we report results on 1000 train-points and 1000 test points, balanced in terms of labels. To use MSE loss, the ten categorical labels were one-hot encoded into vector of zeros and one.

Architecture. We used six convolutional layers with ReLU nonlinearity, kernel of size 5×5, stride of 1, no-padding, no-pooling. The number of input channels was three for the input layer and C for the subsequent five CNN layers. We then vectorized the outputs of the final layer and fed it into an ReLU activated fully connected layer with $25C$ outputs, which were fed into a linear layer with ten outputs corresponding to the ten categories. The loss we used was MSE loss.

Training. Training was carried using full-batch SGD (GD) at varying learning-rates around 5×10^{-4}, Gaussian white noise was added to the gradients to generate $\sigma^2 = 0.2$ in the NNGP-correspondence, layer-dependent weight decay and bias decay which implies a (normalized by width) weight variance and bias variance of $\sigma_w^2 = 2$ and $\sigma_b^2 = 1$, respectively, when trained with no-data. During training we saved, every 1000 epochs, the outputs of the CNN on every test point. We note in passing that the standard deviation of the test outputs around their training-time-averaged value was about 0.1 per CNN output. Training was carried for around half a million epochs which enabled us to reach a statistical error of about 2×10^{-4}, in estimating the mean-squared-discrepancy between the training-time-averaged CNN outputs and our NNGP predictions. Notably our best agreement between the DNN and GP occurred at 112 channels where the MSE was about 7×10^{-3}. Notably the variance of the CNN (the average of its outputs squared) with no data, was about 25.

Statistics. To train our CNN within the regime of the NNSP correspondence, sufficient training time (namely, epochs) was needed to get estimates of the average outputs $\bar{f}_E(x_\alpha) = \bar{f}(x_\alpha) + \delta f_\alpha$ since the estimators' fluctuations, δf_α, scale as $(\tau/t_{\text{training}})^{-1/2}$, where τ is an autocorrelation time scale. Notably, apart from just random noise when estimating the relative MSE between the averaged CNN outputs and the GP, a bias term appears equal to the variance of δf_α averaged over all α's as indeed

$$\sum_{\alpha=1}^{n_{\text{test}}} [\bar{f}_E(x_\alpha) - f_{\text{GP}}(x_\alpha)]^2 = \sum_{\alpha=1}^{n_{\text{test}}} [\bar{f}(x_\alpha) - f_{\text{GP}}(x_\alpha)]^2$$

$$- 2 \sum_{\alpha=1}^{n_{\text{test}}} [\bar{f}_E(x_\alpha) - f_{\text{GP}}(x_\alpha)] \delta f_\alpha$$

$$+ \sum_{\alpha=1}^{n_{\text{test}}} (\delta f_\alpha)^2. \tag{G1}$$
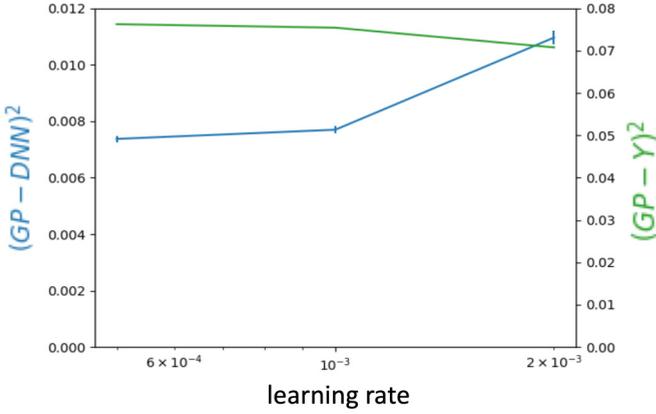
FIG. 6. MSE between our CNN with $C = 48$ and its NNGP as a function of three learning rates.

In all our experiments this bias was the dominant source of statistical error. One can estimate it roughly given the number of uncorrelated samples taken into $\bar{f}_E(x_\alpha)$ and correct the estimator. We did not do so in the main text to make the data analysis more transparent. Since the relative MSEs go down to 7×10^{-3} and the fluctuations of the outputs quantified by $\Sigma_\alpha = (\delta f_\alpha)^2$ are of the order 0.1^2, the amount of uncorrelated samples of CNN outputs we require should be much larger than $0.1^2/(7 \times 10^{-3}) \approx 1.43$. To estimate this bias in practice we repeated the experiment with 3–7 different initialization seeds and deduced the bias from the variance of the results. For comparison with NNGP (our DNN-GP plots) the error bars were proportional to the variance of δf_α. For comparison with the target, we took much larger error bars equal to the uncertainty in estimating the expected loss from a test set of size 1000. These latter error bars where estimated empirically by measuring the variance across ten smaller test sets of size 100.

Last, we discarded the initial "burn-in" epochs, where the network has not yet reached equilibrium. We took this burn-in time to be the time it takes the train-loss to reach within 5% of its stationary value at large times. We estimated the stationary values by waiting until the DNNs train loss remained constant (up to trends much smaller than the fluctuations) for about 5×10^5 epochs. This also coincided well with having more or less stationary test loss.

Learning rate. To be in the regime of the NNSP correspondence, the learning rate must be taken small enough such that discrepancy resulting from having discretization correction to the continuum Langevin dynamics falls well below those coming from finite-width. We find that higher C require lower learning rates, potentially due to the weight decay term being large at large width. In Fig. 6 we report the relative MSE between the NNGP and CNN at learning rates of 0.002, 0.001, 0.0005 and $C = 48$ showing good convergence already at 0.001. Following this we used learning rates of 0.0005 for $C \leqslant 48$ and 0.00025 for $C > 48$, in the main figure.

Comparison with the NNGP. Following Novak *et al.* [14], we obtained the Kernel of our CNN. Notably, since we did not have pooling layers this can be done straightforwardly without any approximations. The NNGP predictions were then obtained in a standard manner [3].

APPENDIX H: FURTHER NUMERICAL RESULTS ON CNNs

Here we report two additional numerical results following the CNN experiment we carried out (for details see Appendix G). Figure 7(b) is the same as Fig. 7(a), apart from the fact that we subtracted our estimate of the statistical bias of our MSE estimator described in Appendix G.

Concerning the experiment with 10 training points. Here we used the same CNN as in the previous experiment. The noise level was again the same and led to an effective $\sigma^2 = 0.1$ for the GP. The weight decay on the biases was taken to be ten times larger leading to $\sigma_b^2 = 0.1$ instead of $\sigma_b = 1.0$ as before. For $C \leqslant 80$ we used a learning rate of $\eta = 5 \times 10^{-5}$ after verifying that reducing it further had no appreciable effect. For $C \leqslant 80$ we used $\eta = 2.5 \times 10^{-5}$. For $c \leqslant 80$ we used $6 \times 10^{+5}$ training epochs and we averaged over 4 different initialization seeds. For $C > 80$ we used between 10 and 16 different initialization seeds. We reduced the aforementioned statistical bias in estimating the MSE from all our MSEs. This bias, equal to the variance of the averaged outputs, was estimated based on our different seeds. The error bars equal this estimated variance which was the dominant source of error.

APPENDIX I: THE FOURTH CUMULANT CAN DIFFERENTIATE CNNs FROM LCNs

Here we show that while the NNGP kernel K of a CNN without pooling cannot distinguish a CNN from an LCN, the fourth cumulant, U, can. For simplicity let us consider the simplest CNN without pooling consisting of the following parts: (1) a 1D image with one color and channel (X_i) as input $i \in \{0, \ldots, L - 1\}$; (2) a single convolutional layer with some activation ϕ acting with stride 1 and no-padding using the conv-kernel T_x^c where $c \in \{1, \ldots, C\}$ is a channel number index and $x \in \{0, \ldots, 2l\}$ is the relative position in the image. Notably, in an LCN this conv-kernel will receive an additional dependence on $\tilde{x}$, the location on X_i on which the kernel acts; (3) a vectorizing operation taking the C outputs of each convolutional around a point $\tilde{x} \in \{l, \ldots, L - l\}$, into a single index $y \in \{0, \ldots, C(L - 2l)\}$; (4) a linear fully connected layer with weights $W_{c\tilde{x}}^o$ where $o \in \{0, \ldots, \#\text{outputs}\}$ are the output indices.

Consider first the NNGP of such a random DNN with weights chosen according to some iid Gaussian distribution $P_0(w)$, with w including both $W_{c\tilde{x}}^o$ and T_x^c. Denoting by $z^o(x)$ the o'th output of the CNN, for an input x we have (where we denote in this section $\langle \cdots \rangle := \langle \cdots \rangle_{P_0(w)}$)

$$
\begin{aligned}
K^{oo'}(x, x') &\equiv \langle z^o(x) z^{o'}(x') \rangle \\
&= \delta_{oo'} \sum_{c,c',\tilde{x},\tilde{x}'} \langle W_{c\tilde{x}}^o W_{c'\tilde{x}'}^{o'} \rangle \langle \phi(T_x^c(\tilde{x}) X_{x+\tilde{x}-l}) \\
&\quad \times \phi(T_x^{c'}(\tilde{x}') X_{x+\tilde{x}'-l}) \rangle.
\end{aligned}
\tag{I1}
$$

The NNGP kernel of an LCN is the same as that of a CNN. This stems from the fact that $\langle W_{c\tilde{x}}^o W_{c'\tilde{x}'}^o \rangle$ yields a Kronecker δ function on the $\tilde{x}, \tilde{x}'$ indices. Consequently, the difference between LCN and CNN, which amounts to whether $T_x^c(\tilde{x})$ is the same (CNN) or a different (LCN) random variable
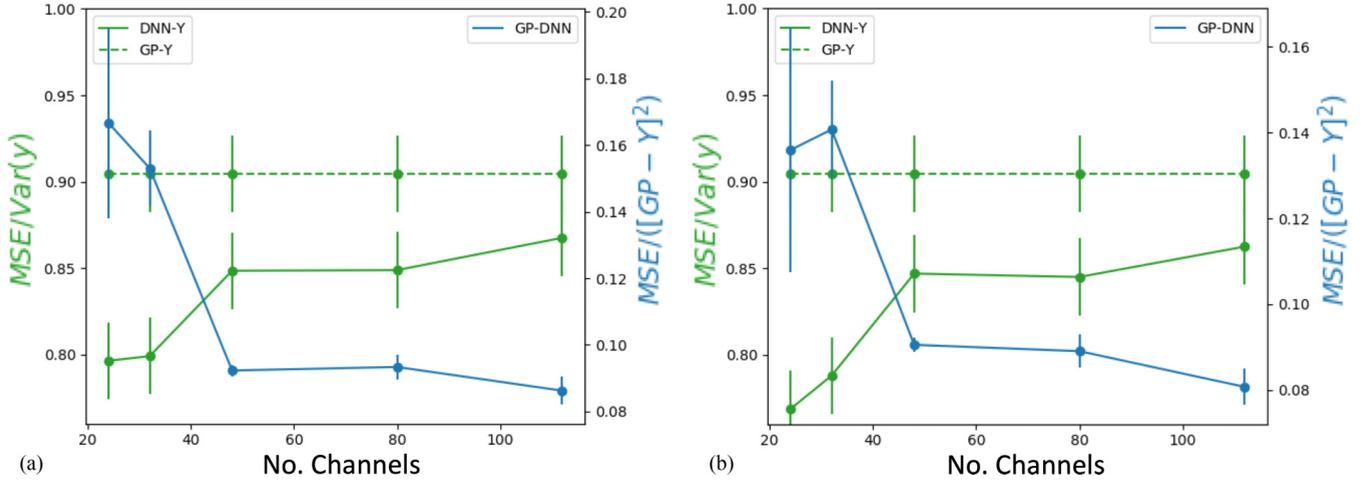
FIG. 7. CNNs trained on CIFAR10 in the regime of the NNSP correspondence compared with NNGPs MSE test loss normalized by target variance of a deep CNN (solid green) and its associated NNGP (dashed green) along with the MSE between the NNGP's predictions and CNN outputs normalized by the NNGP's MSE test loss (solid blue, and on a different scale). We used balanced training and test sets of size 1000 each. For the largest number of channels we reached, the slope of the discrepancy between the CNN's GP and the trained DNN on the log-log scale was -1.77, placing us close to the perturbative regime where a slope of -2.0 is expected. Error bars here reflect statistical errors related only to output averaging and not due to the random choice of a test-set. The performance *deteriorates* at large $N = \#$Channels as the NNSP associated with the CNN approaches an NNGP.

than $T^c_{x' \neq x}(\tilde{x}')$, becomes irrelevant as the these two are never averaged together.

For simplicity, we turn to the fourth cumulant of the same output, given by

$$
\langle z^o(x_1) \cdots z^o(x_4) \rangle - \langle z^o(x_\alpha) z^o(x_\beta) \rangle \langle z^o(x_\gamma) z^o(x_\delta) \rangle [3]
$$
$$
= \langle z^o(x_1) \cdots z^o(x_4) \rangle - K(x_\alpha, x_\beta) K(x_\gamma, x_\delta)[3], \quad \text{(I2)}
$$

with the second term on the left-hand side (LHS) implying all pair-wise averages of $z^o(x_1)..z^o(x_4)$. Note that the first term on the LHS is not directly related to the kernel, thus it has a chance of differentiating a CNN from an LCN. Explicitly, it reads

$$
\sum_{c_1..c_4 \tilde{x}_1..\tilde{x}_4} \langle W^o_{c_1 \tilde{x}_1} \cdots W^o_{c_4 \tilde{x}'_4} \rangle \langle \phi \big(T^{c_1}_{x_1}(\tilde{x}_1) X_{x_1 + \tilde{x}_1 - l} \big) \cdots
$$
$$
\times \phi \big(T^{c_4}_{x_4}(\tilde{x}_4) X_{x_4 + \tilde{x}'_4 - l} \big) \rangle \quad \text{(I3)}
$$

The average over the four W's yields nonzero terms of the type $W^o_{c\tilde{x}} W^o_{c\tilde{x}} W^o_{c'\tilde{x}'} W^o_{c'\tilde{x}'}$ with either $\tilde{x} = \tilde{x}'$ (type 1), $\tilde{x} \neq \tilde{x}'$ and $c \neq c'$ (type 2), or $\tilde{x} \neq \tilde{x}'$ and $c = c'$ (type 3).

The type 1 contribution cannot differentiate an LCN form a CNN since, as in the NNGP case, they always involve only one $\tilde{x}$. The type 2 contribution also cannot differentiate since it yields

$$
\sum_{c \neq c'; \tilde{x} \neq \tilde{x}'} \langle W^o_{c\tilde{x}} W^o_{c\tilde{x}} \rangle \langle W^o_{c'\tilde{x}'} W^o_{c'\tilde{x}'} \rangle \langle \phi \big(T^c_x(\tilde{x}) X_{x + \tilde{x} - l} \big)
$$
$$
\times \phi \big(T^c_x(\tilde{x}) X_{x + \tilde{x} - l} \big) \phi \big(T^{c'}_{x'}(\tilde{x}') X_{x' + \tilde{x}' - l} \big) \phi \big(T^{c'}_{x'}(\tilde{x}') X_{x' + \tilde{x}' - l} \big) \rangle
$$
$$
\text{(I4)}
$$

Examining the average involving the four T's, one finds that since $T^c_x(\tilde{x})$ is uncorrelated with $T^{c'}_{x'}(\tilde{x}')$ for both LCNs and

CNNs, it splits into

$$
\sum_{c \neq c'; \tilde{x} \neq \tilde{x}'} \langle W^o_{c\tilde{x}} W^o_{c\tilde{x}} \rangle \langle W^o_{c'\tilde{x}'} W^o_{c'\tilde{x}'} \rangle \langle \phi \big(T^c_x(\tilde{x}) X_{x + \tilde{x} - l} \big) \phi \big(T^c_x(\tilde{x}) X_{x + \tilde{x} - l} \big) \rangle
$$
$$
\times \langle \phi \big(T^{c'}_{x'}(\tilde{x}') X_{x' + \tilde{x}' - l} \big) \phi \big(T^{c'}_{x'}(\tilde{x}') X_{x' + \tilde{x}' - l} \big) \rangle \quad \text{(I5)}
$$

where as in the NNGP, two T's with different $\tilde{x}$ are never averaged together and we only get a contribution proportional to products of two K's. We note in passing that these type 2 terms yield a contribution that largely cancels that of $K(x_\alpha, x_\beta) K(x_\gamma, x_\delta)[3]$, apart from a "diagonal" contribution ($\tilde{x} = \tilde{x}'$).

We turn our attention to the type 3 term given by

$$
\sum_{c; \tilde{x} \neq \tilde{x}'} \langle W^o_{c\tilde{x}} W^o_{c\tilde{x}} \rangle \langle W^o_{c\tilde{x}'} W^o_{c\tilde{x}'} \rangle \langle \phi \big(T^c_x(\tilde{x}) X_{x + \tilde{x} - l} \big)
$$
$$
\times \phi \big(T^c_x(\tilde{x}) X_{x + \tilde{x} - l} \big) \phi \big(T^c_{x'}(\tilde{x}') X_{x' + \tilde{x}' - l} \big) \phi \big(T^c_{x'}(\tilde{x}') X_{x' + \tilde{x}' - l} \big) \rangle
$$
$$
\text{(I6)}
$$

Examining the average involving the four T's, one now finds a sharp difference between an LCN and a CNN. For an LCN, this average would split into a product of two K's since $T^c_x(\tilde{x})$ would be uncorrelated with $T^c_x(\tilde{x}')$. For a CNN however, $T^c_x(\tilde{x})$ is the same random variable as $T^c_x(\tilde{x}')$ and therefore the average does not split giving rise to a distinct contribution that differentiates a CNN from an LCN. Notably, it is small by a factor of $1/C$ owing to the fact that it contains a redundant summation over one c-index while the averages over the four W's contain a $1/C^2$ factor when properly normalized.

APPENDIX J: BACKGROUND ON THE EQUIVALENT KERNEL (EK)

In this Appendix we generally follow Ref. [3], see also Ref. [45] for more details. The posterior mean for GP regression 6 can be obtained as the function which minimizes the

functional

$$J[f] = \frac{1}{2\sigma^2} \sum_{\alpha=1}^{n} (y_\alpha - f(x_\alpha))^2 + \frac{1}{2} \|f\|_{\mathcal{H}}^2, \quad \text{(J1)}$$

where $\|f\|_{\mathcal{H}}$ is the RKHS norm corresponding to kernel K. Our goal is now to understand the behavior of the minimizer of $J[f]$ as $n \to \infty$. Let the data pairs (x_α, y_α) be drawn from the probability measure $\mu(x, y)$. The expectation value of the MSE is

$$\mathbb{E}\left[\sum_{\alpha=1}^{n} (y_\alpha - f(x_\alpha))^2\right] = n \int (y - f(x))^2 d\mu(x, y) \quad \text{(J2)}$$

Let $g(x) \equiv \mathbb{E}[y|x]$ be the ground truth regression function to be learned. The variance around $g(x)$ is denoted $\sigma^2(x) = \int (y - g(x))^2 d\mu(y|x)$. Then writing $y - f = (y - g) + (g - f)$ we find that the MSE on the data target y can be broken up into the MSE on the ground truth target g plus variance due to the noise

$$\int (y - f(x))^2 d\mu(x, y)$$
$$= \int (g(x) - f(x))^2 d\mu(x) + \int \sigma^2(x) d\mu(x) \quad \text{(J3)}$$

Since the right term on the RHS of Eq. (J3) does not depend on f we can ignore it when looking for the minimizer of the functional which is now replaced by

$$J_\mu[f] = \frac{n}{2\sigma^2} \int (g(x) - f(x))^2 d\mu(x) + \frac{1}{2} \|f\|_{\mathcal{H}}^2. \quad \text{(J4)}$$

To proceed we project g and f on the eigenfunctions of the kernel with respect to $\mu(x)$ which obey $\int \mu(x') K(x, x') \psi_s(x') = \lambda_s \psi_s(x)$. Assuming that the kernel is nondegenerate so that the ψ's form a complete orthonormal basis, for a sufficiently well-behaved target we may write $g(x) = \sum_s g_s \psi_s(x)$, where $g_s = \int g(x) \psi_s(x) d\mu(x)$, and similarly for f. Thus, the functional becomes

$$J_\mu[f] = \frac{n}{2\sigma^2} \sum_s (g_s - f_s)^2 + \frac{1}{2} \sum_s \frac{f_s^2}{\lambda_s}. \quad \text{(J5)}$$

This is easily minimized by taking the derivative w.r.t. each f_s to yield

$$f_s = \frac{\lambda_s}{\lambda_s + \sigma^2/n} g_s. \quad \text{(J6)}$$

In the limit $n \to \infty$ we have $\sigma^2/n \to 0$, thus we expect that f would converge to g. The rate of this convergence will depend on the smoothness of g, the kernel K and the measure $\mu(x, y)$. From Eq. (J6) we see that if $n\lambda_s \ll \sigma^2$ then f_s is effectively zero. This means that we cannot obtain information about the coefficients of eigenfunctions with small eigenvalues until we get a sufficient amount of data. Plugging the result Eq. (J6) into $f(x) = \sum_s f_s \psi_s(x)$ and recalling $g_s = \int g(x') \psi_s(x') d\mu(x')$ we find

$$f(x) = \sum_s \frac{\lambda_s g_s}{\lambda_s + \sigma^2/n} \psi_s(x)$$
$$= \int \underbrace{\sum_s \frac{\lambda_s \psi_s(x) \psi_s(x')}{\lambda_s + \sigma^2/n}}_{h(x,x')} g(x') d\mu(x'). \quad \text{(J7)}$$

This is Eq. (13) from the main text. The term $h(x, x')$ it the *equivalent kernel*. Notice the similarity to the vector-valued equivalent kernel weight function $\mathbf{h}(x_*) = (\mathbf{K} + \sigma^2 I)^{-1} \mathbf{k}(x_*)$ where $\mathbf{K}$ denotes the $n \times n$ matrix of covariances between the training points with entries $K(x_\mu, x_\nu)$ and $\mathbf{k}(x_*)$ is the vector of covariances with elements $K(x_\mu, x_*)$. The difference is that in the usual discrete formulation the prediction was obtained as a linear combination of a finite number of observations y_i with weights given by $h_i(x)$ while here we have instead a continuous integral.

APPENDIX K: CORRECTIONS TO EK

Here we derive finite-N correction to the equivalent kernel result. Using the tools developed by Cohen *et al.* [4], the replicated partition function relevant for estimating the predictions of the network $[f(x_*)]$ averaged ($\langle \cdots \rangle_n$) over all draws of datasets of size n' with n' taken from a Poisson distribution with mean n is given by

$$Z_n = \int \mathcal{D}f e^{-S_{\text{GP}}[f] - \frac{n}{2\sigma^2} \int d\mu_x (f(x) - y(x))^2} (1 + S_U[f])$$
$$+ O(1/N^2), \quad \text{(K1)}$$

with $S_{\text{GP}}[f]$ and $S_U[f]$ given in Eq. (8). We comment that the above expression is only valid for obtaining the leading-order asymptotics in n. Enabling generic n requires introducing replicas explicitly (see Cohen *et al.* [4]). Notably, the above expression coincides with that used for a finite dataset, with two main differences: all the sums over the training set have been replaced by integrals with respect to the measure, μ_x, from which data points are drawn. Furthermore, σ^2 is now accompanied by n. Following this, all the diagrammatic and combinatorial aspects shown in the derivation for a finite dataset hold here as well. For instance, let us examine a specific contribution coming from the quartic term in $H[f]$: $U_{x_1..x_4} K_{x_1 x_1'}^{-1} \cdots K_{x_4 x_4'}^{-1} f(x_1') \cdots f(x_4')$, and from the diagram/Wick-contraction where we take the expectation value of 3 of the 4 f's in this quartic term, to arrive at an expression which is ultimately cubic in the targets y,

$$U_{x_1,x_2,x_3,x_4} K_{x_1 x_1'}^{-1} \langle f(x_1') \rangle_\infty K_{x_2 x_2'}^{-1} \langle f(x_2') \rangle_\infty K_{x_3 x_3'}^{-1} \langle f(x_3') \rangle_\infty K_{x_4 x_4'}^{-1}$$
$$\times \Sigma_\infty(x_4', x_*), \quad \text{(K2)}$$

where we recall that $\langle f(x) \rangle_\infty = K_{xx'} \tilde{K}_{x'x''}^{-1} y(x'')$ and $\Sigma_\infty(x_1, x_2) = K_{x_1, x_2} - K_{x_1, x'} \tilde{K}_{x', x''}^{-1} K_{x'', x_2}$ being the posterior covariance in the EK limit, where $\tilde{K}_{xx'} f(x') = K_{xx'} f(x') + (\sigma^2/n) f(x)$. Using the fact that $K_{xx'}^{-1} K_{x'x''}$ gives a δ function w.r.t. the measure, the integrals against $K_{x_\alpha x_\alpha'}^{-1}$ can be easily carried out yielding

$$\left(U_{x_1,x_2,x_3,x_*} - U_{x_1,x_2,x_3,x_4} \tilde{K}_{x_4,x_4'}^{-1} K_{x_4',x_*}'\right) \tilde{K}_{x_1,x_1'}^{-1} \tilde{K}_{x_2,x_2'}^{-1} \tilde{K}_{x_3,x_3'}^{-1} y(x_1') y$$
$$\times (x_2') y(x_3'). \quad \text{(K3)}$$

Introducing the discrepancy operator $\tilde{\delta}_{xx''} := \delta_{xx''} - K_{xx'} \tilde{K}_{x'x''}^{-1} = \frac{\sigma^2}{n} \tilde{K}_{xx''}^{-1}$, we can write a more compact expression,

$$\left(\frac{n}{\sigma^2}\right)^3 \tilde{\delta}_{x_*,x_4} U_{x_1,x_2,x_3,x_4} \tilde{\delta}_{x_1,x_1'} \tilde{\delta}_{x_2,x_2'} \tilde{\delta}_{x_3,x_3'} y(x_1')(x_2') y(x_3'). \quad \text{(K4)}$$

This with the additional 1/4! factor times the combinatorial factor of 4 related to choosing the "partner" of $f(x_*)$ in the Wick contraction, yields an overall factor of 1/6 as in the main text, Eq. (14). The other term therein, which is linear in y, is a result of following similar steps with the $\bar{f} \Sigma \Sigma_*$ contributions that do not get canceled by the quadratic part in $H[f]$.

---

[1] A. Daniely, R. Frostig, and Y. Singer, Toward deeper understanding of neural networks: The power of initialization and a dual view on expressivity, Adv. Neural Inf. Syst. **29**, 2253 (2016).

[2] A. Jacot, F. Gabriel, and C. Hongler, Neural tangent kernel: Convergence and generalization in neural networks, arXiv:1806.07572 (2018).

[3] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning: Adaptive Computation and Machine Learning* (MIT Press, Cambridge, MA, 2005).

[4] O. Cohen, O. Malka, and Z. Ringel, Learning curves for overparametrized deep neural networks: A field theory perspective, Phys. Rev. Research **3**, 023034 (2021).

[5] N. Rahaman, A. Baratin, D. Arpit, F. Draxler, M. Lin, F. A. Hamprecht, Y. Bengio, and A. Courville, On the spectral bias of neural networks, ICML **97**, 5301 (2019).

[6] R. Basri, D. Jacobs, Y. Kasten, and S. Kritchman, The convergence rate of neural networks for learned functions of different frequencies, arXiv:1906.00425 (2019).

[7] L. Chizat, E. Oyallon, and F. Bach, On lazy training in differentiable programming, in *Advances in Neural Information Processing Systems* (MIT Press, Cambridge, MA, 2019), pp. 2937–2947.

[8] M. Geiger, L. Petrini, and M. Wyart, Landscape and training regimes in deep learning, Phys. Rep. **924**, 1 (2021).

[9] S. Arora, S. S. Du, W. Hu, Z. Li, R. Salakhutdinov, and R. Wang, On exact computation with an infinitely wide neural net, arXiv:1904.11955 (2019).

[10] E. Dyer and G. Gur-Ari, Asymptotics of wide networks from feynman diagrams, in *Proceedings of the International Conference on Learning Representations* (ICLR, 2020).

[11] J. Huang and H.-T. Yau, Dynamics of deep neural networks and neural tangent hierarchy, arXiv:1909.08156 (2019).

[12] J. Lee, J. Sohl-dickstein, J. Pennington, R. Novak, S. Schoenholz, and Y. Bahri, Deep neural networks as Gaussian processes, in *Proceedings of the International Conference on Learning Representations* (2018).

[13] A. G. d. G. Matthews, M. Rowland, J. Hron, R. E. Turner, and Z. Ghahramani, Gaussian process behaviour in wide deep neural networks, arXiv:1804.11271 (2018).

[14] R. Novak, L. Xiao, J. Lee, Y. Bahri, G. Yang, D. A. Abolafia, J. Pennington, and J. Sohl-Dickstein, Bayesian Deep convolutional networks with many channels are Gaussian processes, arXiv:1810.05148 (2018).

[15] W. Maddox, T. Garipov, P. Izmailov, D. Vetrov, and A. G. Wilson, A simple baseline for Bayesian uncertainty in deep learning, arXiv:1902.02476 (2019).

[16] S. Mandt, M. D. Hoffman, and D. M. Blei, Stochastic gradient descent as approximate Bayesian inference, J. Mach. Learn. Research **18**, 1 (2017).

[17] Y. W. Teh, A. H. Thiery, and S. J. Vollmer, Consistency and fluctuations for stochastic gradient Langevin dynamics, J. Mach. Learn. Res. **17**, 193 (2016).

[18] M. Welling and Y. W. Teh, Bayesian learning via stochastic gradient Langevin dynamics, in *Proceedings of the 28th International Conference on International Conference on Machine Learning (ICML'11)* (Omnipress, 2011), pp. 681–688.

[19] N. Ye, Z. Zhu, and R. K. Mantiuk, Langevin dynamics with continuous tempering for training deep neural networks, arXiv:1703.04379 (2017).

[20] B. Hanin and M. Nica, Finite depth and width corrections to the neural tangent kernel, arXiv:1909.05989 (2019).

[21] S. Yaida, Non-Gaussian processes and neural networks at finite widths, in *Mathematical and Scientific Machine Learning* (PMLR, 2020).

[22] Z. Chen, Y. Cao, Q. Gu, and T. Zhang, Mean-field analysis of two-layer neural networks: Non-asymptotic rates and generalization bounds, arXiv:2002.04026 (2020).

[23] S. Mei, A. Montanari, and P.-M. Nguyen, A mean-field view of the landscape of two-layer neural networks, Proc. Natl. Acad. Sci. USA **115**, E7665 (2018).

[24] B. Tzen and M. Raginsky, A mean-field theory of lazy training in two-layer neural nets: Entropic regularization and controlled McKean-Vlasov dynamics, arXiv:2002.01987 (2020).

[25] D. Araújo, R. I. Oliveira, and D. Yukimura, A mean-field limit for certain deep neural networks, arXiv:1906.00193 (2019).

[26] P.-M. Nguyen, Mean-field limit of the learning dynamics of multilayer neural networks, arXiv:1902.02880 (2019).

[27] R. M. Neal, MCMC using Hamiltonian dynamics, in *Handbook of Markov Chain Monte Carlo*, edited by S. Brooks, A. Gelman, G. Jones, and X. L. Meng (Chapman & Hall/CRC Press, 2011).

[28] H. Risken and T. Frank, *The Fokker-Planck Equation: Methods of Solution and Applications*, Springer Series in Synergetics (Springer, Berlin, 1996).

[29] L. S. Schulman, *Techniques and Applications of Path Integration* (Courier Corporation, 2012).

[30] Y. Cho and L. K. Saul, Kernel methods for deep learning, in *Proceedings of the 22nd International Conference on Neural Information Processing Systems (NIPS'09)* (Curran Associates, Red Hook, NY, 2009), pp. 342–350.

[31] We take the *total error*, i.e., we do not divide by n so that $\mathcal{L}[f]$ becomes more dominant for larger n.

[32] Here σ^2 is a property of the training protocol and not of the data itself, or our prior on it.

[33] E. Gardner and B. Derrida, Optimal storage properties of neural network models, J. Phys. A: Math. Gen. **21**, 271 (1988).

[34] F. Draxler, K. Veschgini, M. Salmhofer, and F. A. Hamprecht, Essentially no barriers in neural network energy landscape, in *Proceedings of the 35th International Conference on Machine Learning* (PMLR, 2018), Vol. 80, pp. 1308–1317.

[35] L. Bottou, Stochastic gradient descent tricks, in *Neural Networks: Tricks of the Trade* (Springer, Berlin, 2012), pp. 421–436.

[36] E. Hoffer, I. Hubara, and D. Soudry, Train longer, generalize better: Closing the generalization gap in large batch training of

neural networks, *Advances in Neural Information Processing Systems 30* (2017), pp. 1729–1739.

[37] A. Lewkowycz, Y. Bahri, E. Dyer, J. Sohl-Dickstein, and G. Gur-Ari, The large learning rate phase of deep learning: The catapult mechanism, arXiv:2003.02218 (2020).

[38] S. L. Smith, P.-J. Kindermans, C. Ying, and Q. V. Le, Don't decay the learning rate, increase the batch size, arXiv:1711.00489 (2017).

[39] C. Mingard, G. Valle-Pérez, J. Skalse, and A. A. Louis, Is SGD a Bayesian sampler? Well, almost, J. Mach. Learn. Res. **22**, 1 (2021).

[40] S. L. Smith, B. Dherin, D. G. Barrett, and S. De, On the origin of implicit regularization in stochastic gradient descent, arXiv:2101.12176 (2021).

[41] P. McCullagh, *Tensor Methods in Statistics: Monographs on Statistics and Applied Probability* (Chapman and Hall/CRC, 2018).

[42] E. Sellentin, A. H. Jaffe, and A. F. Heavens, On the use of the Edgeworth expansion in cosmology I: How to foresee and evade its pitfalls, arXiv:1709.03452 (2017).

[43] S. d'Ascoli, M. Refinetti, G. Biroli, and F. Krzakala, Double trouble in double descent: Bias and variance(s) in the lazy regime, in *Proceedings of the International Conference on Machine Learning* (PMLR, 2020), pp. 2280–2290.

[44] M. Geiger, A. Jacot, S. Spigler, F. Gabriel, L. Sagun, S. d'Ascoli, G. Biroli, C. Hongler, and M. Wyart, Scaling description of generalization with number of parameters in deep learning, J. Stat. Mech. (2020) 023401.

[45] P. Sollich and C. K. Williams, Understanding Gaussian process regression using the equivalent kernel, in *Proceedings of the International Workshop on Deterministic and Statistical Methods in Machine Learning* (Springer, Berlin, 2004), pp. 211–228.

[46] J. Lee, S. S. Schoenholz, J. Pennington, B. Adlam, L. Xiao, R. Novak, and J. Sohl-Dickstein, Finite versus infinite neural networks: An empirical study, arXiv:2007.15801 (2020).

[47] B. Neyshabur, Z. Li, S. Bhojanapalli, Y. LeCun, and N. Srebro, Towards understanding the role of over-parametrization in generalization of neural networks, arXiv:1805.12076 (2018).

[48] A. Zee, *Quantum Field Theory in a Nutshell*, Nutshell handbook (Princeton University Press, Princeton, NJ, 2003).

[49] P. A. P. Moran, Rank correlation and product-moment correlation, Biometrika **35**, 203 (1948).