Decomposing neural networks as mappings of correlation functions

Kirsten Fischer,[1,2,*] Alexandre René,[1,3,4] Christian Keup,[1,2] Moritz Layer,[1,2] David Dahmen,[1] and Moritz Helias[1,4]

[1]*Institute of Neuroscience and Medicine (INM-6) and Institute for Advanced Simulation (IAS-6) and JARA-Institute Brain Structure-Function Relationships (INM-10), Jülich Research Centre, 52425 Jülich, Germany*

[2]*RWTH Aachen University, 52062 Aachen, Germany*

[3]*Department of Physics, University of Ottawa, K1N 6N5 Ottawa, Canada*

[4]*Department of Physics, Faculty 1, RWTH Aachen University, 52074 Aachen, Germany*

(Received 24 February 2022; accepted 19 September 2022; published 28 November 2022)

Understanding the functional principles of information processing in deep neural networks continues to be a challenge, in particular for networks with trained and thus nonrandom weights. To address this issue, we study the mapping between probability distributions implemented by a deep feed-forward network. We characterize this mapping as an iterated transformation of distributions, where the nonlinearity in each layer transfers information between different orders of correlation functions. This allows us to identify essential statistics in the data, as well as different information representations that can be used by neural networks. Applied to an XOR task and to MNIST, we show that correlations up to second order predominantly capture the information processing in the internal layers, while the input layer also extracts higher-order correlations from the data. This analysis provides a quantitative and explainable perspective on classification.

DOI: 10.1103/PhysRevResearch.4.043143

I. INTRODUCTION

Recent years have shown a great success of deep neural networks in solving a wide range of tasks, from image recognition [1] to playing Go [2]. One major branch is supervised learning, where input-output mappings are learned from examples. In many common problems the target output values are given by a finite set, defining a classification task [3]. The objective then is to minimize an error measure between the correct class label and the prediction made by the neural network with respect to the joint probability distribution of data samples and class labels [4]. Thus, training dynamics, and consequently the solution strategy implemented by the network, depend on this probability distribution and the information it encodes. In this view, a network implements a transformation of the input distribution with the objective to concentrate the output distribution around the assigned target values for each class. How such a transformation is achieved and how the network training depends on the statistics of the presented data is, however, still mostly unknown.

To render the decision-making process of neural networks transparent, a profound understanding regarding their functional principles and extraction of meaningful features from given data is required. Over the past years the discrepancy between success in applications and limited understanding has led to an increased interest also in the theoretical community [4–9]. An important line of theoretical research investigates ensembles of neural networks in the limit of infinite width for which the central limit theorem implies an exact equivalence to Gaussian processes (GPs) [10–13]. While this approach is informative with respect to how relations between data samples are transformed by the network, it does not reveal how the internal structure of data samples is processed. As an example, for image classification the Gaussian process view takes into account the relation between all corresponding pixels x_i^α, x_i^β of any pair of images α, β in the form of a scalar product $\sum_i x_i^\alpha x_i^\beta$. Even though the data statistics shape the eigenfunctions of the GP's covariance matrix [14], Sec. 4.3, it is not obvious which role is played by the structure within individual images determined by, e.g., correlations between pixel values x_i^α and x_j^α. In particular, due to the rotational symmetry of the scalar product, the GP view gives identical results when image pixels are shuffled consistently across all images. However, clearly the internal structure of data samples also contains important information that may be employed to solve a given task. The focus of this study is to investigate how this information is extracted from the data and utilized by the network to perform classification.

Other approaches [15–17], similarly as GPs, focus on ensembles of networks with randomly drawn weights. In contrast, here we study how particular realizations of trained and untrained networks process different statistical features of the data. Thus, we shift the perspective from distributions over network parameters to distributions over the data. In particular, we describe the input-output mapping implemented

*ki.fischer@fz-juelich.de

by deep neural networks in terms of correlation functions. To trace the transformation of correlation functions across layers of neural networks, we make use of methods from statistical field theory [18–21] and obtain recursive relations in a perturbative manner by means of Feynman diagrams. Our results yield a characterization of the network as a nonlinear mapping of correlation functions, where each layer exchanges information between different statistical orders. Reexpressing the loss function in terms of data correlations allows us to study their role in the training process, and to link the transformation of data correlations to the solution strategies found by the network. For the particular example of the mean squared error loss function, we show that network training relies exclusively on the first two cumulants of the output (mean and covariance), while these, in turn, are predominantly determined by means and covariances of network activations in previous layers. Furthermore, we show that corrections from higher-order correlations to mean and covariance, which are readily computable with the proposed generic field-theoretical framework, are of greatest importance in the first layer, where these corrections effectuate the information flow from higher-order correlations to mean and covariance.

The structure of this study is as follows: Section II provides theoretical background on the definition and architecture of deep neural networks (Sec. II A), on empirical risk minimization in the context of classification (Sec. II B), and on field-theoretical descriptions of probability distributions in terms of cumulants and their generating function (Sec. II C). In Sec. III we decompose the network mapping into correlation functions, tracing their transformations backwards through the network. We start by relating the loss to the first- and second-order correlations of the network outputs (Sec. III A), then discuss the mapping of correlations by individual hidden layers (Sec. III B), and end with the extraction of data correlations by the input layer (Sec. III C). Section IV applies these theoretical tools to several example data sets. We start with an adaptation of the XOR problem, where the input statistics are fully known and selectively presented to the network to study different encoding and processing schemes of class identities (Sec. IV B). We proceed with an application to the MNIST data set [22], where we show that classification performance is largely based on the transformation of means and covariances across layers (Sec. IV C). Finally, we showcase the importance of higher-order correlations and their extraction in the input layer by constructing a data set where information on class identity is only encoded in correlations of third and higher order (Sec. IV D). In Sec. V we discuss our results and provide an outlook.

II. THEORETICAL BACKGROUND

A. Feed-forward network architecture

We consider fully connected neural networks with L layers of N_l neurons each, and one additional linear readout layer, as shown in Fig. 1(a). Each layer $l = 1, \ldots, L$ consists of an affine transformation

$$z_i^l = \sum_{j=1}^{N_{l-1}} W_{ij}^l y_j^{l-1} + b_i^l \tag{1}$$

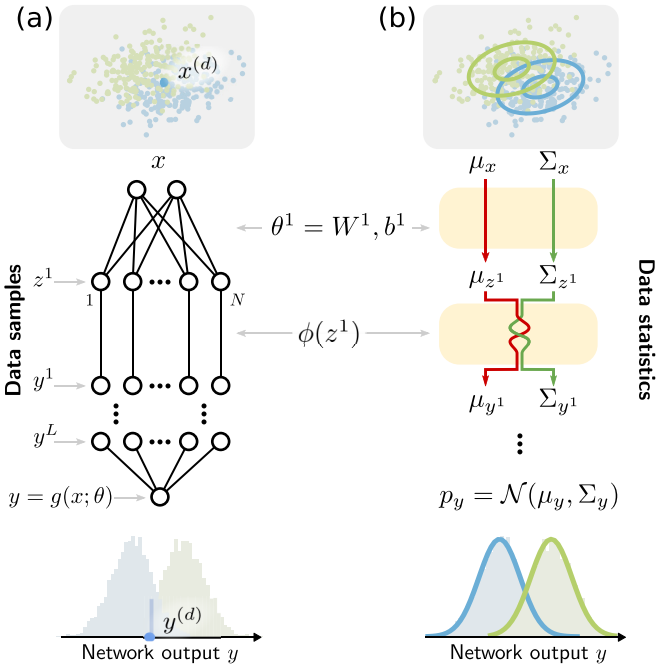


FIG. 1. (a) Network analysis based on data samples considers each data sample $x^{(d)}$ separately as it passes through the network, producing a single corresponding output $y^{(d)}$. Each layer consists of an affine transformation (W^l, b^l) followed by a nonlinearity ϕ applied componentwise. (b) Network analysis based on data statistics considers how the entire data distribution $p(x)$ is transformed by the network. At each step, the intermediate distribution is parametrized by its cumulants, the most important of which are the mean μ and the covariance Σ. The affine step transforms μ and Σ independently, while the nonlinearity ϕ causes a nontrivial interaction of the two.

parametrized by a weight matrix $W^l \in \mathbb{R}^{N_l \times N_{l-1}}$ and bias vector $b^l \in \mathbb{R}^{N_l}$. This step is followed by the pointwise application of a nonlinear activation function ϕ, yielding

$$y_i^l = \phi(z_i^l) = \phi\left(\sum_{j=1}^{N_{l-1}} W_{ij}^l y_j^{l-1} + b_i^l\right). \tag{2}$$

Here $y^0 = x \in \mathbb{R}^{N_0}$ denotes the input data of dimension N_0. The readout layer produces the network output $y \in \mathbb{R}^{d_{\text{out}}}$, specifically, $y_i = z_i^{L+1}$. The network mapping $y = g(x;\theta)$ is given by iterating over network layers and characterized by parameters $\theta := \{W^l, b^l\}_{l=1,\ldots,L+1}$.

We initialize all network parameters randomly from independent and identically distributed (i.i.d.) centered Gaussians $W_{ij}^l \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, \sigma_w^2/N_{l-1})$ and $b_i^l \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, \sigma_b^2)$ before training. The scaling of the variance σ_w^2/N_{l-1} is chosen such that the covariance of z^l [see Eq. (1)] is independent of the layer width.

B. Learning theory: Empirical risk minimization

The fundamental assumption underlying classification[1] is the existence of a joint distribution $p(x, t)$ of data samples x

[1]Since this work focuses on classification tasks, we tailor the presentation of empirical risk minimization to that context.

and class labels t that is the same for training and evaluation [3]. By Bayes' theorem, the distribution of the input data can be treated as a mixture model $p(x) = \sum_t p(t)\, p(x|t)$. The network's task is then to implement a mapping $g : x \mapsto y$ that minimizes the expectation of a loss $\ell(y, t)$ between the network outputs $y = g(x; \theta)$ and the labels t.

This mapping, in turn, induces a mapping of the probability distributions

$$p(x|t) \mapsto p(y|t;\theta) = \int \delta[y - g(x;\theta)]\, p(x|t)\, dx \tag{3}$$

for each label t, where $\delta(\cdot)$ refers to the Dirac delta distribution. The unconditioned output distribution is then the weighted sum $p(y) = \sum_t p(t)\, p(y|t;\theta)$. Ideally, the network output y matches the true label t so that the target distribution is given by $p(y|t) = \delta(y - t)$.

Training algorithms seek to minimize the expected loss or risk functional [23]

$$R(\theta) = \langle \ell(y,t) \rangle_{y|\theta} = \sum_t p(t)\, \langle \ell(y,t) \rangle_{y|t;\theta}, \tag{4}$$

where the expectation value $\langle \cdot \rangle_{y|t;\theta}$ is taken with regard to the class-conditional output distributions $p(y|t;\theta)$. In general, neither the mixture components of the input distribution $p(x|t)$ nor the induced class-conditional output distributions $p(y|t;\theta)$ are known. Instead, the expected loss is replaced by the empirical loss or risk

$$R_{\text{emp}}(\theta) = \frac{1}{D} \sum_{d=1}^{D} \ell\big(g(x^{(d)};\theta), t^{(d)}\big), \tag{5}$$

evaluated for a training set $\{(x^{(d)}, t^{(d)})\}_d$, with D being its size and d the respective sample index. The *empirical risk minimization principle* then assumes the following: the mapping $g(\cdot\,;\theta^*)$ that minimizes the empirical risk $\theta^* = \text{argmin}_\theta\, R_{\text{emp}}(\theta)$ yields an expected risk $R(\theta^*)$ that is close to its minimum $\min_\theta R(\theta)$ [23].

C. Parametrization of probability distributions in terms of cumulants

This section contains the framework to track data correlations (cumulants) of arbitrary order through the network. Large parts of the main text deal with the first two orders, the Gaussian approximation. Readers who want to obtain an overview of the main results may skip the remainder of this section at first read. This part, however, becomes essential when including non-Gaussian corrections and as a means to obtain an intuitive picture of how nonlinear transformations couple the different statistical orders.

Neural networks can be regarded as complex systems that generate many interactions between data components. A common approach to investigate such systems is by studying generating functions of moments or cumulants rather than the probability distributions themselves. Cumulants often provide a more convenient parametrization of probability distributions as they are additive with respect to addition of independent variables, leading to simpler expressions for the transformation of statistics across layers. Such approaches are common in statistical physics and in mathematical statistics.

The network mapping $g : x \mapsto y$ relates the cumulant generating function of network outputs y to the statistics of the input x:

$$\mathcal{W}_{y|t;\theta}(j) = \ln \langle \exp(j^\intercal y) \rangle_{y|t;\theta} \tag{6}$$

$$= \ln \langle \exp(j^\intercal g(x;\theta)) \rangle_{x|t}. \tag{7}$$

The cumulant generating function is considered per class t as the data statistics are expected to differ between classes. The class-conditional output cumulant of order n denoted by $G^{(n)}_{y|t;\theta}$ is then defined as

$$G^{(n)}_{y|t;\theta} = \left. \frac{d^n \mathcal{W}_{y|t;\theta}(j)}{dj^n} \right|_{j=0}.$$

Evaluating Eq. (7) would in principle allow one to relate $G^{(n)}_{y|t;\theta}$ to the input cumulants $G^{(n')}_{x|t;\theta}$. However, one intricacy is that the network mapping $g(x;\theta)$ is given via the iterations in Eq. (2). Their iterative nonlinear nature makes deep neural networks powerful as universal function approximators, but complicates their analysis in terms of data processing. Yet, we can study the transformation of cumulants from input to output by considering layers individually.

Since preactivations z^l are determined by affine linear transformations, the cumulant generating function of preactivations z^l in layer l is trivially related to the cumulant generating function of postactivations y^{l-1} of layer $l-1$ as

$$\begin{aligned} \mathcal{W}_{z^l}(j) &= \ln \langle \exp(j^\intercal z^l) \rangle_{z^l} \\ &= \ln \langle \exp(j^\intercal W^l\, y^{l-1} + j^\intercal b^l) \rangle_{y^{l-1}} \\ &= \mathcal{W}_{y^{l-1}}((W^l)^\intercal j) + j^\intercal b^l, \end{aligned} \tag{8}$$

yielding for the first-order cumulant ($n = 1$)

$$G^{(1)}_{z^l} = W^l\, G^{(1)}_{y^{l-1}} + b^l, \tag{9}$$

and for second- and higher-order cumulants ($n \geqslant 2$)

$$G^{(n)}_{z^l,\,(r_1,\ldots,r_n)} = \sum_{s_1,\ldots,s_n} W^l_{r_1 s_1} \ldots W^l_{r_n s_n}\, G^{(n)}_{y^{l-1},\,(s_1,\ldots,s_n)}. \tag{10}$$

Each index s_i is hence contracted with one factor $W^l_{r_k s_i}$ to produce the index r_k of the resulting cumulant. Consequently, cumulants of preactivations z^l are linear tensor transformations of cumulants of postactivations y^{l-1} of the same order.

The nonlinear activation function ϕ in each layer l then relates the preactivations z^l to the corresponding postactivations y^l:

$$\begin{aligned} \mathcal{W}_{y^l}(j) &= \ln \langle \exp(j^\intercal y^l) \rangle_{y^l} \\ &= \ln \langle \exp(j^\intercal \phi(z^l)) \rangle_{z^l}. \end{aligned} \tag{11}$$

This cumulant generating function of the postactivations y^l cannot, in general, be computed exactly. One common approximation technique is a perturbative expansion [21], which we here recast in the following way: by replacing $\phi(z^l)$ with its Taylor expansion $\sum_m \frac{\phi^{(m)}|_{z^l=0}}{m!} (z^l)^m$ in Eq. (11) and treating

TABLE I. Diagrammatic elements for the perturbative expansion of $\mathcal{W}_{y^l}(j)$ for $y^l = \phi(z^l)$.

Meaning	Algebraic term	Graphical representation
External line	$j_r\,\delta_{rs}$	$j_r \quad z^l_s$
Cumulant vertex with n internal lines	$G^{(n)}_{z^l,(r_1,\ldots,r_n)}$	$z^l_{r_1}$, $z^l_{r_n}$ $\ldots$
ϕ-vertex with m internal lines and one external line	$j_r \frac{1}{m!}\phi^{(m)}\big\vert_{z^l=0} \times \delta_{ri_1}\ldots\delta_{ri_m}$	j_r, $z^l_{i_1}$, $\ldots$, $z^l_{i_m}$

nonlinear terms ($m > 1$) as perturbations, we can construct cumulants $G^{(n)}_{y^l}$ as series of Feynman diagrams composed of the graphical elements shown in Table I. For example, for the mean of the first layer we get the following diagrams:

$$G^{(1)}_{y^1,i} = \quad + \quad + \ldots$$

$$= \frac{\phi^{(1)}\big\vert_{x=0}}{1!} G^{(1)}_{x,i} + \frac{\phi^{(2)}\big\vert_{x=0}}{2!} G^{(2)}_{x,ii} + \ldots . \tag{12}$$

We find that in general these expressions involve two types of factors, which we represent with two types of vertices: empty circles with internal lines, representing cumulants $G^{(n)}_{z^l}$ of preactivations z^l, and hatched circles with one external line j that stem from Taylor coefficients $\frac{\phi^{(m)}\big\vert_{z^l=0}}{m!}$ of the nonlinearity.

For constructing a cumulant $G^{(n)}_{y^l}$ of the postactivations y^l of order n, we need to determine all diagrams with n external lines. External lines occur on cumulant vertices as well as on hatched vertices. Furthermore, they always need to be connected to a cumulant vertex, but cannot be connected to one another. Finally, due to the *linked cluster theorem*, only connected diagrams need to be considered since others do not contribute to cumulants. When evaluating the generated diagrams, all permutations of indices $(r_1, \ldots, r_n)$ for both internal and external lines need to be taken into account. Symmetries within diagrams result in their repeated occurrence, which is reflected in combinatorial prefactors (for more details, see [21]).

Using this perturbative approach for determining the cumulants $G^{(n)}_{y^l}$ of the postactivations y^l has two main advantages: First, it provides a principled way to go beyond Gaussian statistics and include higher-order cumulants. Second, the availability of a diagrammatic language allows us to graphically represent the information transfer from cumulants $G^{(n)}_{z^l}$ of the preactivations z^l to cumulants $G^{(m)}_{y^l}$ of the postactivations y^l.

The diagrammatic representation introduced above assumes that the activation function ϕ can be expanded as a Taylor series. For nondifferentiable functions such as ReLU, this approach can be adapted by using a Gram-Charlier expansion of the probability distribution $p(z^l)$. The expectation value in Eq. (11) then becomes a sum of Gaussian integrals, which can be calculated either analytically (see Appendix C for ReLU as an example) or numerically.

III. DECOMPOSING DEEP NEURAL NETWORKS INTO CORRELATION FUNCTIONS

Analyzing how deep networks process data is difficult due to their iterative, parameter-dependent definition. Statistical learning theory studies the expected error [24], thus shifting from the transformation of data samples to that of data distributions. We follow this idea here by studying how data correlations of the input are iteratively transformed by deep networks, as illustrated in Fig. 1, and how they shape the expected loss.

A. Data correlations drive network training

We here discuss the dependence of the expected loss in Eq. (4) on the data correlations. In general, the expected risk $R(\theta)$ is a function of the class labels t and the class-conditional cumulants $G^{(n)}_{y|t;\theta}$ of arbitrary orders n:

$$\begin{aligned} R(\theta) &= \sum_t \int dy\, \ell(y,t)\, p(y|t;\theta) \\ &=: \sum_t \sigma_t\left(\left\{G^{(n)}_{y|t;\theta}\right\}_n\right) \\ &=: \sigma\left(\left\{\left\{G^{(n)}_{y|t;\theta}\right\}_n;t\right\}_t\right). \end{aligned}$$

However, for the often employed mean-squared error $\ell_{\mathrm{MSE}}(y,t) = \|y-t\|^2$, $R(\theta)$ only depends on the mean μ^t_y and variance Σ^t_y of outputs of each class t as

$$R_{\mathrm{MSE}}\left(\left\{\mu^t_y, \Sigma^t_y; t\right\}_t\right) = \sum_t p(t)\left(\mathrm{tr}\,\Sigma^t_y + \|\mu^t_y - t\|^2\right). \tag{13}$$

Training therefore aims to match class means and labels, while minimizing the variance of each class's output.[2]

In this case, the first- and second-order cumulants (mean and covariance) of the last layer alone drive network training, thus singling these out as the relevant statistics. This result has two implications: (1) In deep feed-forward networks, only non-Gaussian statistics that appear in network layers before the final layer can contribute to the learned information processing by influencing the first two cumulants in the final layer. (2) If networks produce non-Gaussian statistics in the final layer, these do not serve a functional role per se; rather they may arise as a by-product of earlier layers operating on higher-order statistics.

[2]Equation (13) should not be confused with the bias-variance decomposition [25], where the expectation over finite data sets of fixed size is taken instead of the expectation over the input distribution $p(x)$ itself.

Thus, understanding the network mapping reduces to understanding how the Gaussian statistics (μ_y^t, Σ_y^t) of the output arise from the presented data distribution across multiple network layers. Network training and the resulting information processing within the network is therefore directly linked to how data correlations are transformed by the network.

B. Propagation of data correlations within the network

To understand how the extraction of information from the input and its internal processing shape the first- and second-order cumulants of the output, we follow these two quantities backwards through the network. According to Eqs. (8)–(10), the affine transformation in each layer implies for the preactivations z^l

$$\mu_{z^l} = W^l\,\mu_{y^{l-1}} + b^l, \quad \Sigma_{z^l} = W^l\,\Sigma_{y^{l-1}}(W^l)^\mathsf{T}, \tag{14}$$

showing that the two quantities are transformed independently of each other in this step (Fig. 1).

In general, the nonlinear activation function $\phi : z^l \mapsto y^l$ makes the statistics of the postactivations y^l dependent on cumulants of arbitrary orders in the preactivations z^l through (cf. Sec. II C)

$$\mu_{y^l} = \left.\frac{d\mathcal{W}_{y^l}(j)}{dj}\right|_{j=0} = \langle\phi(z^l)\rangle_{z^l}, \tag{15}$$

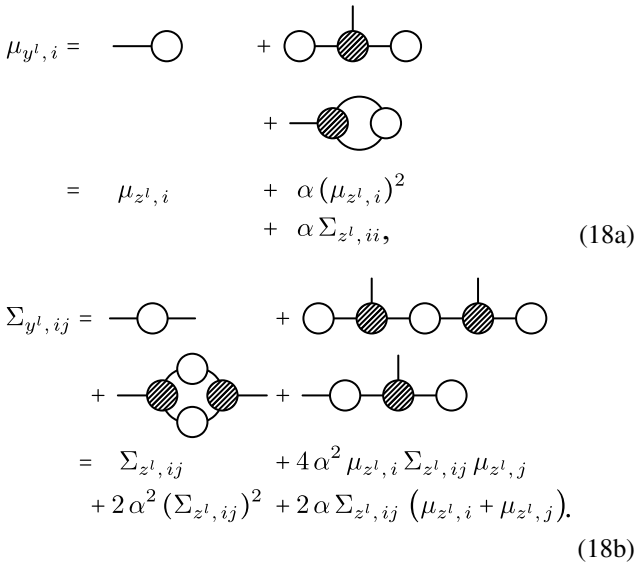
$$\Sigma_{y^l} = \left.\frac{d^2\mathcal{W}_{y^l}(j)}{dj\,dj^\mathsf{T}}\right|_{j=0} = \langle\phi(z^l)\,\phi(z^l)^\mathsf{T}\rangle_{z^l} - \mu_{y^l}\mu_{y^l}^\mathsf{T}. \tag{16}$$

However, due to the central limit theorem, initializing the weights independently causes the affine transformation $y^{l-1} \mapsto z^l$ to mainly pass on the Gaussian part of the statistics since higher-order cumulants $G^{(n)}_{z^l,(i_1,\ldots,i_n)} = \langle\!\langle z^l_{i_1} z^l_{i_2} \ldots z^l_{i_n}\rangle\!\rangle \sim O[(N_{l-1})^{-\frac{n}{2}+1}]$ are suppressed by the layer width N_{l-1} for $n > 2$. In Appendix B we derive sufficient conditions under which the Gaussian approximation remains valid also for wide trained networks. In brief, we find that it suffices to have a natural scaling of weights $w \sim O(N^{-\frac{1}{2}})$ as well as an approximate orthogonal decomposition of the sending layer's covariance matrix by the row vectors of the connectivity to the next layer, Eq. (B7). These conditions are in particular different from those of the lazy (kernel or neural tangent kernel) regimes, where weights only change marginally. Under these conditions, in the limit of infinitely wide networks, expectations over preactivations $\langle\cdot\rangle_{z^l}$ can be taken with respect to Gaussian distributions $z^l \sim \mathcal{N}(\mu_{z^l}, \Sigma_{z^l})$, and we obtain that the mean and covariance of postactivations are nonlinear functions of only mean and covariance of preactivations

$$\mu_{y^l} = f_\mu(\mu_{z^l}, \Sigma_{z^l}), \quad \Sigma_{y^l} = f_\Sigma(\mu_{z^l}, \Sigma_{z^l}). \tag{17}$$

These functions mediate interactions between first- and second-order cumulants.

Applying this argument iteratively to the network layers $l = L+1, L, \ldots, 2$, it follows that the information processing in the *internal* network layers is largely determined by an iterated, nonlinear mapping of mean and covariance. The interaction functions f_μ and f_Σ can be calculated numerically for arbitrary activation functions. In particular, ϕ need not be differentiable. Analytic expressions can be obtained for various activation functions ϕ; we provide expressions for $\phi = \mathrm{ReLU}$ and $\phi(z) = z + \alpha\,z^2$ in Appendix C, Table II. The latter, minimally nonlinear activation function yields especially interpretable interaction functions that are constructed from the following diagrams:



The last diagram contributing to $\Sigma_{y^l,ij}$ corresponds to an expression containing two terms. These terms result from the permutation of the indices (i, j) (see Sec. II C).

Training introduces correlations between weights, thus violating the independence assumption of the central limit theorem. Also, the sufficient conditions for the Gaussian approximation to be consistent (Appendix B) are not necessary conditions; for example, pairs of neurons may be perfectly correlated without violating a Gaussian description. We will therefore show in the following that empirically the first- and second-order cumulants provide a useful approximation for the information propagation *within* the network.

C. Information extraction in the input layer

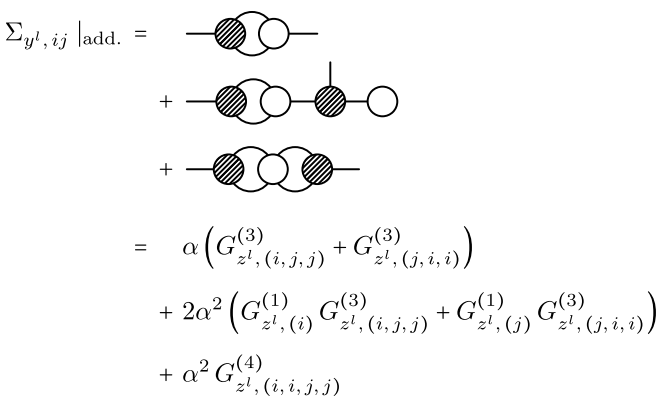
So far we have studied the internal network layers. Here, we discuss the role of the input layer in extracting information from higher-order correlations of the input data. Since the preactivations of this layer $z^1_i = \sum_{j=1}^{N_0} W^1_{ij}\,x_j + b^1_i$ involve a sum over the input dimension N_0 instead of the network width N, higher-order cumulants $G^{(n>2)}_{z^1}$ scale with $N_0^{1-\frac{n}{2}}$ and need to be taken into account for smaller input dimension N_0. In consequence, cumulants of multiple orders n contribute to the mean and covariance of the postactivations y^1:

$$\mu_{y^1} = h_\mu\big(\{G^{(n)}_{z^1}\}_n\big), \quad \Sigma_{y^1} = h_\Sigma\big(\{G^{(n)}_{z^1}\}_n\big). \tag{19}$$

These mean and covariance are then passed on through the entire network.

The interaction functions h_μ and h_Σ can be systematically approximated for any activation function, either by the diagrammatic techniques discussed in Sec. II C in the case of differentiable functions or alternatively by a Gram-Charlier expansion for nondifferentiable functions (see Appendix C for ReLU as an example). Analytically simple and exact expressions can be computed for a quadratic nonlinearity [see Eq. (18a)]; in this case, the expression for the mean does not get any contribution from $G^{(n>2)}_{z^1}$, while the covariance gets

additional contributions from third- and fourth-order input correlations:



$$= \alpha\left(G^{(3)}_{z^l,(i,j,j)} + G^{(3)}_{z^l,(j,i,i)}\right) + 2\alpha^2\left(G^{(1)}_{z^l,(i)}\, G^{(3)}_{z^l,(i,j,j)} + G^{(1)}_{z^l,(j)}\, G^{(3)}_{z^l,(j,i,i)}\right) + \alpha^2\, G^{(4)}_{z^l,(i,i,j,j)}$$

As in the previous section, there are two diagrams that each correspond to an expression containing multiple terms. These terms result from the permutation of the indices (i, j) (see Sec. II C).

The cumulants of the preactivations $G^{(n)}_{z^1}$ are linked to the cumulants of the input data $G^{(n)}_x$ by a mapping between corresponding orders n as $G^{(n)}_x \to G^{(n)}_{z^1}$ [see Eq. (10)], yielding

$$\mu_{y^1} = \tilde{h}_\mu\left(\left\{G^{(n)}_x\right\}_n; \{W^1, b^1\}\right),$$
$$\Sigma_{y^1} = \tilde{h}_\Sigma\left(\left\{G^{(n)}_x\right\}_n; \{W^1, b^1\}\right).$$

Thus, the input layer effectively extracts information from higher-order correlations of the input data $G^{(n>2)}_x$.

D. Statistical model of a feed-forward network

Putting together all previous sections, we introduce the *statistical model* corresponding to a given network model. This model represents the information processing performed by the network in terms of the data correlations $\{G^{(n)}_x\}_n$.

By iterating Eqs. (14), (19), and (17), respectively, across layers, one obtains the mean and covariance of the network output $y = g(x;\theta)$ as functions of the statistics of x:

$$\mu_y = W^{L+1}\left(f_\mu\left(\ldots\left\{\tilde{h}_\nu\left(\left\{G^{(n)}_x\right\}_n\right)\right\}_{\nu=\mu,\Sigma}\ldots\right)\right) + b^{L+1} =: g_\mu\left(\left\{G^{(n)}_x\right\}_n;\theta,\phi\right), \tag{20}$$

$$\Sigma_y = W^{L+1}\left(f_\Sigma\left(\ldots\left\{\tilde{h}_\nu\left(\left\{G^{(n)}_x\right\}_n\right)\right\}_{\nu=\mu,\Sigma}\ldots\right)\right)(W^{L+1})^{\mathrm{T}} =: g_\Sigma\left(\left\{G^{(n)}_x\right\}_n;\theta,\phi\right). \tag{21}$$

Since the network decomposes into a mapping for each class label t, one obtains the distribution of the network output as a Gaussian mixture $p(y) = \sum_t p(t)\,\mathcal{N}(\mu^t_y, \Sigma^t_y)(y)$. The parameters (μ^t_y, Σ^t_y) are determined by the propagation of data correlations $\{G^{(n),t}_x\}_{n,t}$ through the network equations (20) and (21). Note that these are generally not exact due to the Gaussian approximation of preactivations z^l at each intermediate layer. In the following, we call the mapping

$$g_{\mathrm{stat}} : \left(\left\{G^{(n),t}_x\right\}_{n,t}, \theta, \phi\right) \mapsto p(y) \tag{22}$$

the *statistical model* of the network. One important feature is that the statistical model shares the parameter structure $\theta = \{W^l, b^l\}_{l=1,\ldots,L+1}$ with the corresponding network model. In consequence, there is a one-to-one correspondence between the statistical model (22) and the network model $g : (x;\theta) \mapsto y$ given a fixed set of parameters θ.

Beyond empirically comparing these two models, the statistical model can be used to assess the relevance of data correlations $\{G^{(n),t}_x\}_{n,t}$ for solving a particular task. We have shown in Sec. III A that the expected mean squared error loss function $R_{\mathrm{MSE}}(\{\mu^t_y, \Sigma^t_y\}_t)$ is given by Eq. (13), so that it depends solely on mean and covariance of the output. By the statistical model (22), the mean-squared error R_{MSE} thus can be approximated as a function of the data correlations $\{G^{(n),t}_x\}_{n,t}$ and the network parameters θ:

$$R_{\mathrm{MSE}}\left(\left\{\mu^t_y, \Sigma^t_y\right\}_t\right) \tag{23}$$

$$\approx R_{\mathrm{MSE}}\left(\left\{G^{(n),t}_x\right\}_{n,t};\theta\right). \tag{24}$$

Minimizing this loss then yields optimal parameters θ^* for the statistical model. The corresponding network model $g(x;\theta^*)$ is then dependent on the given set of data correlations $\{G^{(n),t}_x\}_{n,t}$, allowing the investigation of their relevance in solving a particular network task.

IV. EXPERIMENTAL RESULTS

We now apply the developed methods to the XOR problem and the MNIST data set. We use the network architecture defined in Sec. II A with fixed network width $N_l = N$ for $l \geqslant 1$ and either the ReLU activation function or a minimal nonlinearity, namely, the quadratic activation function $\phi(z) = z + \alpha\, z^2$ with $\alpha = 0.5$.

A. Training details

For initialization of network parameters θ, we use $\sigma_w^2 = \sigma_b^2 = 0.75$. Following the standard procedure, networks are trained by optimizing the empirical risk per data batch $\{(x^{(b)}, t^{(b)})\}_b$ of the expected MSE loss:

$$R_{\mathrm{emp,MSE}}(\theta) = \frac{1}{B}\sum_{b=1}^{B} \ell_{\mathrm{MSE}}\left(g\left(x^{(b)};\theta\right), t^{(b)}\right). \tag{25}$$

The batch size B is set to 10 on XOR and 100 on MNIST. For optimization, we use ADAM [26,27] with learning rate 10^{-3}, momenta $\beta_1 = 0.9$ and $\beta_2 = 0.999$, $\epsilon = 10^{-8}$, and $\lambda = 0$. The choice of the optimizer does not affect the above presented derivations. Network implementations were done in PYTORCH [28].

B. Multiple information encodings of the XOR problem

We first study an adaptation of the XOR problem as a nonlinearly separable Gaussian mixture distribution. We make use of two conceptual advantages of this XOR task: First, knowing the exact input distribution allows us to focus on the internal information processing within the network. Second, the fact that each class is itself a mixture distribution allows us to trace the class-conditional correlations in two alternative forms, corresponding to two different statistical representations of class membership, which isolate different statistics of the input, respectively, the mean and the covariance. We find that while the task can be solved for both representations,

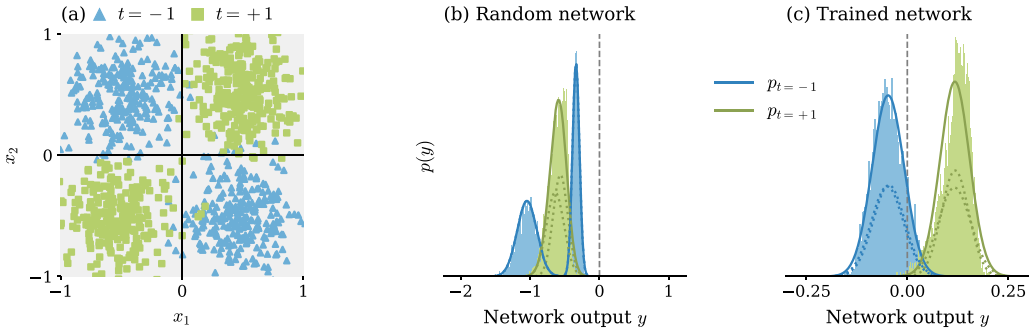


FIG. 2. Information propagation in ReLU networks for the XOR problem. (a) The distribution of input data is modeled as a Gaussian mixture. Data samples $x^{(d)}$ (blue and green dots) are assigned to class labels $t = \pm 1$ based on the respective mixture component. (b), (c) Distribution of the network output for random (b) and trained (c) parameters. Class-conditional distributions (solid curves) are determined as a superposition of the propagated mixture components (dashed curves) as in Eq. (26) and empirical estimates (blue and green histograms) are obtained from the test data. Since networks are trained on class labels $t = \pm 1$, the classification threshold is set to $y = 0$ (gray lines). Other parameters: $\phi =$ ReLU, depth $L = 1$, width $N = 10$; trained network in (c) achieves $P = 93.82\%$ performance compared to $P_{\text{opt}} = 97.5\%$.

they correspond to different local minima of the empirical loss landscape.

1. Problem setup as a Gaussian mixture

Our adaptation of the XOR problem uses real-valued instead of binary inputs and describes the input distribution as a Gaussian mixture of four components, illustrated in Fig. 2(a). For the class label $t = +1$, we choose the mean values of its two components $\pm$ as $\mu_x^{t=+1,\pm} = \pm(0.5, 0.5)^\intercal$; for $t = -1$, we use $\mu_x^{t=-1,\pm} = \pm(-0.5, 0.5)^\intercal$. Covariances are isotropic throughout with $\Sigma_x^{t,\pm} = 0.05\,\mathbb{I}$ and the input distribution

$$p(x,t) = p(t) \sum_{\pm} p_{\pm} \mathcal{N}\big(\mu_x^{t,\pm}, \Sigma_x^{t,\pm}\big)(x)$$

weighs all components equally $p(t) = p_{\pm} = \frac{1}{2}$. A data sample $x^{(d)}$ is assigned a target label $t^{(d)} \in \{\pm 1\}$ based on the mixture component it is drawn from. From the geometry of the problem follows that the optimal decision boundaries coincide with the axes in data space [Fig. 2(a)], allowing us to calculate the optimal performance $P_{\text{opt}} = 97.5\%$. We use training and test data sets of sizes $n_{\text{train}} = 10^5$ and $n_{\text{test}} = 10^4$, respectively.

2. Accuracy of internal information processing in terms of correlation functions

Given the exact input distribution for this problem, we trace the transformation of mean and covariance predicted by Eqs. (20) and (21) for each mixture component $(t, \pm)$ separately, obtaining

$$p_{\text{theo}}(y) = \sum_t p(t) \sum_{\pm} p_{\pm} \mathcal{N}\big(\mu_y^{t,\pm}, \Sigma_y^{t,\pm}\big)(y), \qquad (26)$$

where $\mu_y^{t,\pm} = g_\mu(\mu_x^{t,\pm}, \Sigma_x^{t,\pm}; \theta, \phi)$ and $\Sigma_y^{t,\pm} = g_\Sigma(\mu_x^{t,\pm}, \Sigma_x^{t,\pm}; \theta, \phi)$ are functions of the input statistics, the network parameters, and depend on the choice of activation function. In Fig. 2 we compare this theoretical result to an empirical estimate of the output distribution $p_{\text{emp}}(y)$, given as a histogram obtained from the test data. We test the validity of the statistical model for both an untrained network with random weight initialization [Fig. 2(b)] and a trained network [Fig. 2(c)].

The untrained network produces an output distribution of complex shape composed of superimposed close-to Gaussian distributions, each corresponding to one component, as shown in Fig. 2(b). Training the network reshapes the output distribution such that the class-conditional distributions $p(y|t)$ become well separated by the threshold at $y = 0$, as shown in Fig. 2(c). The overlap between these two distributions around the threshold corresponds to the classification error. Qualitatively, theory and simulation agree well for both random and trained networks. These results apply for different activation functions ϕ [see Fig. 8 in Appendix D for $\phi(z) = z + \alpha z^2$].

To quantify the alignment of theory and simulation, we compute the Kullback-Leibler divergence D_{KL} between the empirical estimate $p_{\text{emp}}(y)$ and the theoretical result $p_{\text{theo}}(y)$, considering the empirical distribution $p_{\text{emp}}(y)$ as the reference. To account for the variability of output distributions across different network realizations, this quantity is normalized by the entropy H of the empirical distribution $p_{\text{emp}}(y)$, yielding $\hat{D}_{\text{KL}}(p_{\text{emp}} \| p_{\text{theo}}) = D_{\text{KL}}(p_{\text{emp}} \| p_{\text{theo}}) / H(p_{\text{emp}})$.

We average $\hat{D}_{\text{KL}}(p_{\text{emp}} \| p_{\text{theo}})$ across 50 different network realizations, for random [Fig. 3(a)] and trained [Fig. 3(b)]

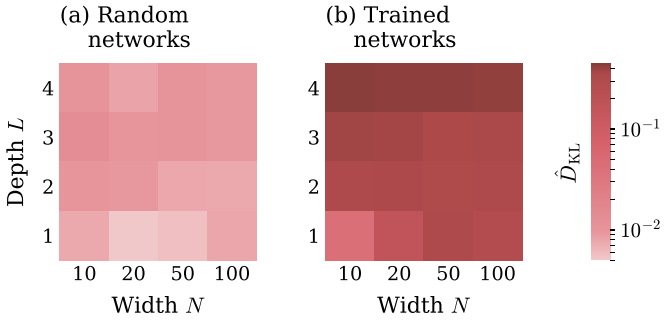


FIG. 3. Deviation between theoretical and empirical output distribution for (a) random and (b) trained networks, measured across 50 different network realizations using the normalized Kullback-Leibler divergence $\hat{D}_{\text{KL}}(p_{\text{emp}} \| p_{\text{theo}})$. On average, the trained networks achieve performance values of $P = 97.00\% \pm 0.05\%$ compared to $P_{\text{opt}} = 97.5\%$. Networks were trained to perform the XOR task described in Sec. IV B 1. Other parameters: $\phi =$ ReLU.

networks. In both cases, the deviation between theory and simulation is generally small, but increases mildly with the network depth L as approximation errors accumulate across network layers. For random networks, the deviations are generally small with a slight decrease of the deviation for wider networks, in agreement with the central limit theorem as discussed in Sec. III B. For trained networks with thus correlated parameters, there is an overall increase of deviations between theory and simulation. Nonetheless, this increase remains modest, showing that the theory continues to be applicable for networks with trained, and thus nonrandom, parameters. Again these results apply for different activation functions ϕ [see Fig. 9 in Appendix D for $\phi(z) = z + \alpha z^2$]. When evaluating the expressions for ReLU, one needs to be careful with the numerics due to the appearing error functions.

3. Different information coding paradigms and their relations

In the previous section, we have shown that the mapping implemented by the network can be described as a mapping of correlation functions [see Eq. (22)]. On the level of data correlations, it directly follows that the network's expressivity with respect to a given task depends on two properties: (1) the ability of the network architecture to implement a desired mapping of data correlations from its input to its output; (2) the way in which information about class membership is represented by data correlations in the input.

A complete study of the first property would be provided by fully describing the space of possible mappings, which is challenging in general. However, the forward mapping of cumulants we have obtained in Sec. III B allows us to probe this space experimentally, and it provides a path to more systematic studies of network expressivity (see our remarks on statistical receptive fields in Sec. V).

In this section, we study the second property by investigating two different information representations: (A) the class membership is represented by different means between classes, while the covariances and all higher-order cumulants are identical; (B) the class membership is represented by different covariances between classes, while the means and all higher-order cumulants are identical. Accordingly, these two representations are called *mean coding* (A) and *covariance coding* (B) in the following. While each of these two settings confines the class membership to one particular cumulant order, the more general case is that class membership is represented by various orders of statistical moments. In that case, the network may make use of this duplicate information to maximize performance.

To be able to compare these settings, in either case we train models on a single task defined via a single data distribution, but present different statistical representations of the data. We use the statistical model corresponding to the network described in Sec. III D, limiting input correlations to mean and covariance by setting higher-order cumulants to zero. We take the binary XOR problem (see Sec. IV B 1) which can be cast into either information representation in a natural way: For mean coding (A), we provide to the network both the class labels t and the specific mixture component $\pm$ from which a sample was drawn, yielding four sets of statistics $\{\mu_x^m,\ \Sigma_x^m\}_{m=(t,\pm)}$ with different means but identical covariances. For covariance coding (B), only the class label t is provided to the network, yielding two sets of statistics $\{\mu_x^m,\ \Sigma_x^m\}_{m=(t)}$, for which the covariances $\Sigma_x^{t=\pm 1} = \left(\begin{smallmatrix} 0.3 & \pm 0.25 \\ \pm 0.25 & 0.3 \end{smallmatrix}\right)$ differ between the two classes, while their means are the same [see Fig. 4(a)]. In both cases, all higher-order cumulants of the component distributions $m = (t, \pm)$ and $m = (t)$, respectively, are set to zero. Note that for mean coding the class distributions $p(x, t = \pm 1) = \sum_\pm p_\pm \mathcal{N}(\mu_x^{t,\pm}, \Sigma_x^{t,\pm})(x)$ indeed include higher-order cumulants. The different sets of input statistics $\{\mu_x^m,\ \Sigma_x^m\}_{m=(t,\pm)}$ (A) and $\{\mu_x^m,\ \Sigma_x^m\}_{m=(t)}$ (B), respectively, define different statistical models for mean and covariance coding.

We compare these two statistical representations A and B of the network to the network trained directly on batches of samples; the latter we refer to as *sample coding* in the following. Sample coding can be considered as the case where potentially all statistical moments of the data are accessible to the network. Our goal is to address the following questions: First, which statistical representation most closely matches the information representation used by a network trained on data samples? Second, is there a difference in performance between information representations; in particular, can the network equivalently use the information provided by either mean or covariance coding? Finally, does the network make use of duplicate information in different cumulant orders to improve performance in the case of sample coding?

To answer these questions, we optimize models until convergence using either representation. We then switch to a different representation, continuing optimization for the same number of steps, and observe the stability of the previously found solution. Each experimental setup is repeated with 10^2 different weight initializations. Results are shown in Fig. 4 for three different coding combinations, where we plot both the loss and the magnitude of change $||\Delta\theta||_2^2$ of the model parameters.

We find that after initial optimization all three models correspond to networks with at least $P = 91\%$ performance, so training converges in all cases and the networks implement viable solutions before the switch. Thus, the behavior after the switch indicates how the found solution is affected by changing the statistical representation. Furthermore, we observe that immediately after the switch from covariance to mean coding, $||\Delta\theta||_2^2$ jumps to values similar to the initial training steps [Fig. 4(b)]. This indicates a near complete change of the model, which suggests that mean and covariance coding induce fundamentally different solutions. In contrast, the jump is modest when switching from covariance to sample coding [Fig. 4(c)] and nonexistent when switching from mean to sample coding [Fig. 4(d)], suggesting that those different solutions coexist in the true loss landscape of the network model. Thus, we find that the network utilizes the presented information in different ways for the two representations, as expected based on the information flow in these networks, derived in Sec. III B.

In particular, the case of covariance coding highlights the importance of a nonlinear activation function when the discriminating information is not contained in the class means. Since classification is based on different mean values in the network output, the difference in covariance for each class

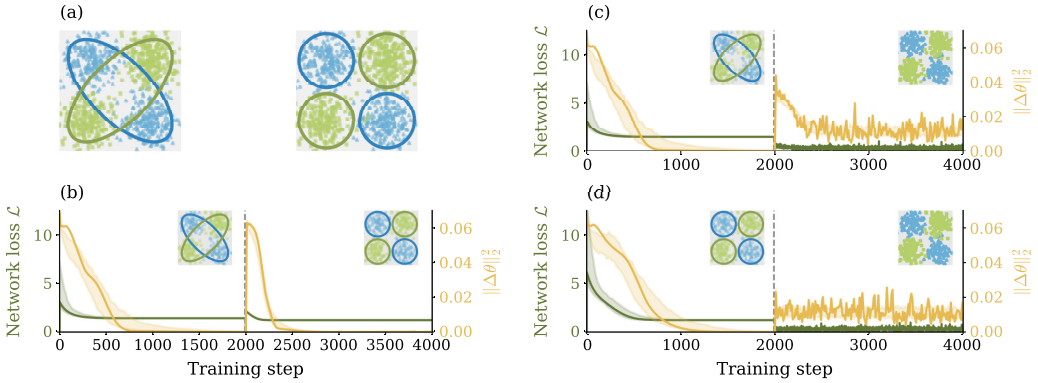


FIG. 4. Mean and covariance coding. (a) Data distributions of XOR task. Left: covariance coding; class membership (blue and green ellipses) is encoded in the covariance alone. Right: mean coding; the two individual Gaussian components of each class (both blue circles and both green circles) differ in their means, while having the same covariance. (b)–(d) Evolution of network loss R [$R_{\text{emp, MSE}}$ Eq. (25) for training the network model, R_{MSE} Eq. (13) for the statistical model] and change of network parameters $||\Delta\theta||_2^2$ in each training step: the first $T_1 = 2000$ training steps train one model, starting from random parameters θ. The model representation is changed at T_1, starting from parameters $\theta(T_1)$ obtained in the preceding period. The change of network parameters is evaluated every 10 training steps $\|\Delta\theta(T)\|_2^2 = \|\theta(T) - \theta(T-10)\|_2^2$. Shaded areas show the typical range, based on lower and upper quartiles across 10^2 network realizations. Solid curves show the behavior of a single network realization. (b) First period: statistical model with covariance coding; second period: statistical model with mean coding. (c) First period: statistical model with covariance coding; second period: network model. (d) First period: statistical model with mean coding; second period: network model. Training parameters: $n_{\text{train}} = 10^4$, 2 epochs. Other parameters: $\phi(z) = z + \alpha z^2$, depth $L = 1$, width $N = 10$.

needs to be transferred to the mean. This information transfer is mediated by the nonlinearity ϕ; for the case $\phi(z) = z + \alpha z^2$ used in Fig. 4, we have the particularly simple transfer function

$$\mu_{y^l,i} = \mu_{z^l,i} + \alpha \left(\mu_{z^l,i}\right)^2 + \alpha \Sigma_{z^l,ii} \tag{27}$$

from covariances to means. Here, only diagonal entries of the covariance enter, while the input covariances $\Sigma_x^{t=\pm1} = \begin{pmatrix} 0.3 & \pm 0.25 \\ \pm 0.25 & 0.3 \end{pmatrix}$ differ in their off-diagonal entries. The information transfer from off-diagonal to diagonal entries is mediated by the affine transformation [see Eq. (14)] prior to the activation function. In this way, we can track how information flows into the mean as it is transformed by successive network layers.

In summary, we find that for this task the network can effectively utilize the information presented by either mean or covariance coding, both representations leading to different solutions with comparable performance. Sample coding tends to yield similar solutions as mean coding, implying that the network makes use of duplicate information present in higher-order moments of the data samples from each class.

C. Essential data correlations of the MNIST data set

We consider in this section the MNIST data set [22], consisting of 10 classes of 28×28 images. This data set is highly structured: if one approximates each class by a multivariate Gaussian, the resulting samples are already visually recognizable [Figs. 5(a) and 5(b); see Appendix E for further details). Our goal is to use the theory developed in previous sections to quantify this observation, in a matter which can be generalized to different data sets and different sets of input cumulants. We also argue that truncation of cumulants in the input layer has the largest impact, and in the process validate our theory on a nontrivial task.

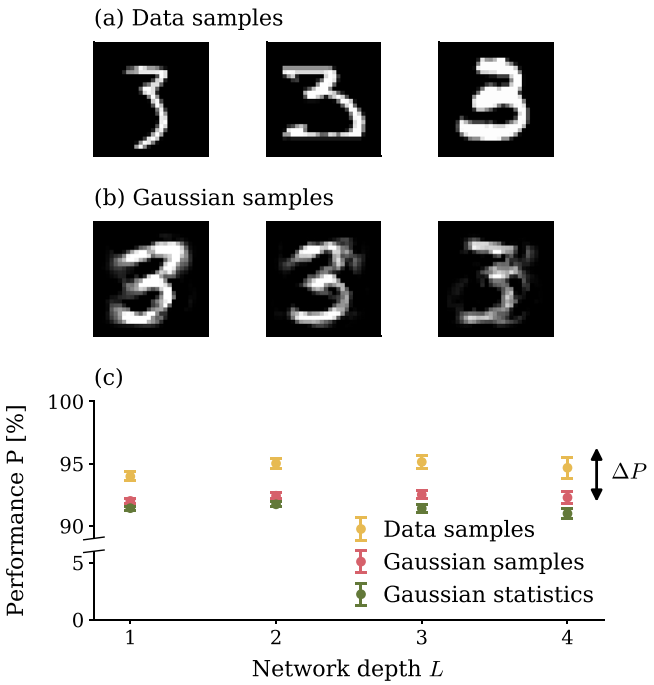


FIG. 5. First- and second-order correlations of MNIST. (a) Three example data samples showing the digit 3 from the MNIST training data set. (b) Data samples showing the digit 3, drawn from the Gaussian approximation of the input distribution. (c) Classification performance on the MNIST test data set for different input encodings. Network training consistently achieves performance values of $P \approx 94\%$ or more (yellow), while performance of the corresponding statistical model is $3.3\% \pm 0.6\%$ lower (green). Training networks on Gaussian input samples yield a comparable performance difference (red). In all cases, performance is evaluated on the MNIST test data set. Error bars show the standard deviation across 10^2 different network realizations. Other parameters: $\phi(z) = z + \alpha z^2$, depth $L \in [1, 2, 3, 4]$, width $N = 100$.

Concretely, we proceed as follows: when optimizing the parameters θ^* of the statistical model, we restrict the data statistics to a particular set of cumulants $\{G_x^{(n)}\}_{n=1,\dots,\hat{n}}$ and compare the achieved performance to that of the network trained on samples $y = g(x;\theta^*)$. The difference in performance is then indicative of the importance of the cumulants we kept. We employ one-hot encoding, making the network output $d_{\text{out}} = 10$ dimensional.

As a baseline, we first train network models on both the MNIST data set [Fig. 5(a)] and the corresponding Gaussian samples [Fig. 5(b)]. The latter case limits the information that can be extracted by the input layer to the class-conditional means μ_x^t and covariances Σ_x^t. In both cases, networks are trained with the standard empirical loss [Eq. (25)]; in particular, this allows inner network layers to make use of cumulants of any order. With respect to classification performance, we find that training on Gaussian samples yields a performance that is lower by $\Delta P \simeq 2.4\% \pm 0.7\%$ [Fig. 5(c)]: a difference we can ascribe to the removal of higher-order cumulants in the data distribution. Based on the modest magnitude of this difference, we conclude that data mean and covariance are already highly informative for these data and account for about $P \approx 91\%$.

We next train the corresponding statistical model [Eq. (22)] on the Gaussian approximation of MNIST [Fig. 5(b)]. Compared to the network model trained on the Gaussian samples corresponding to the same data distribution, we find only slightly lower performance, by about $0.9 \pm 0.4\%$ [Fig. 5(c)], suggesting that the statistical model given by Eq. (22) is a good representation for the information processing in internal network layers. The fact that most of the performance drop with respect to standard training on MNIST is due to the Gaussian approximation of the input data indicates the importance of processing higher-order cumulants by the input layer. In the next section, we show with an illustrative example how these can be included into the theory.

D. Including higher-order correlation functions in the input layer

So far we have studied class-conditional means μ_x^t and covariances Σ_x^t of the input data; however, these two statistics may not always be informative. It is in fact easy to construct a low-dimensional task with two classes $t = \pm 1$, where both class-conditional means and covariances of the data are identical, $\mu_x^{t=-1} = \mu_x^{t=+1}$, $\Sigma_x^{t=-1} = \Sigma_x^{t=+1}$, thereby conveying no information regarding the class membership [Figs. 6(a) and 6(b)]. Classification in such cases must therefore rely on higher-order statistics. For the example in Fig. 6, since third-order cumulants differ between classes ($G_x^{(3),\,t=-1} = -G_x^{(3),\,t=+1}$), we expect their inclusion into the statistical model to be sufficient for solving the task. We here demonstrate that such higher-order cumulants can indeed be treated by our approach; in particular, we validate the statement made in Sec. III C that it suffices to consider higher-order cumulants in only the first layer.

The input distribution for this task is defined as a Gaussian mixture of four components, illustrated in Figs. 6(a) and 6(b) (details in Appendix F). As expected, training the network model yields near-optimal performance values, while a statistical model $g_{\text{stat}}(\{G_x^{(n)}\}_{n=1,2},\theta)$ that considers only the class-conditional means μ_x^t and covariances Σ_x^t fails to solve the task, yielding chance-level performance [Fig. 6(c)]. This performance gap is nearly bridged when we include the third-order input cumulants $G_x^{(3)}$ [via Eq. (C19)] in the first layer of the statistical model $\tilde{g}_{\text{stat}}(\{G_x^{(n)}\}_{n=1,2,3,4},\theta)$. The activation function ϕ allows information in $G_x^{(3)}$ to be transferred to lower-order cumulants, which are then processed by subsequent layers in the manner described in previous sections, facilitating different means $G_y^{(1)}$ in the output of the statistical model.

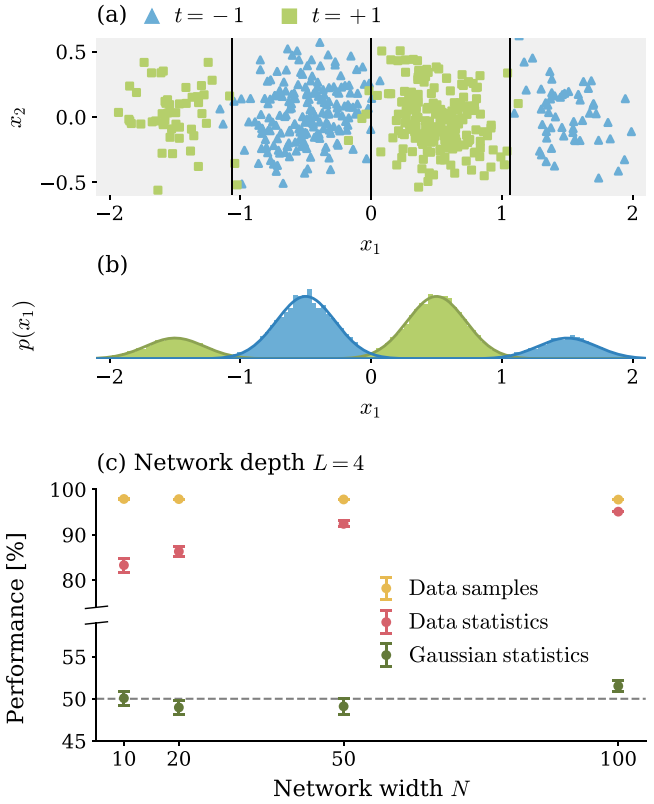


FIG. 6. Information extracted from higher-order correlations. (a) The distribution of input data is modeled as a Gaussian mixture. Data samples $x^{(d)}$ (blue and green dots) are assigned to class labels $t = \pm 1$ based on their respective mixture component. The two classes have zero mean and the same covariance. (b) Projection of data samples to the x_1 axis (histograms), which corresponds to the marginalization of the input distribution with respect to x_2 (solid lines), illustrating the different weighing of the mixture components. (c) Classification performance for different model choices. Network training consistently achieves performance values of $P \approx 96\%$ or more (yellow). Optimizing the statistical model $g_{\text{stat}}(\{G_x^{(n)}\}_{n=1,2},\theta)$ that considers only the first- and second-order correlations (green) results in performance values corresponding to chance level (dotted line). However, including the third-order correlations into the statistical model $\tilde{g}_{\text{stat}}(\{G_x^{(n)}\}_{n=1,2,3,4},\theta)$ nearly bridges this gap (red). In all cases, performance is evaluated on a test data set. Error bars show the error of the mean of performance across 10^2 different network realizations. Other parameters: $\phi(z) = z + \alpha z^2$, depth $L = 4$, width $N = [10,\ 20,\ 50,\ 100]$.

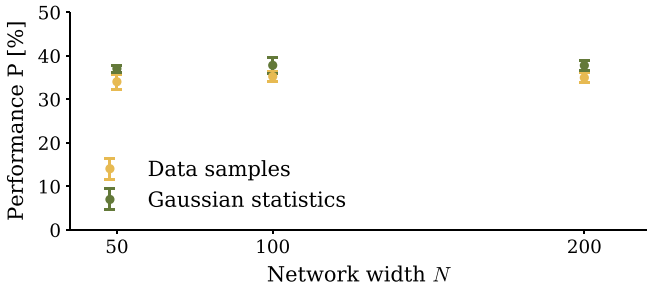


FIG. 7. Classification performance on CIFAR-10. Training the statistical model on Gaussian statistics (green) consistently achieves higher performance values than training network models on data samples (yellow). In both cases, performance is evaluated on the CIFAR-10 test data set. Error bars show the standard deviation across 10 different network realizations. Other parameters: $\phi(z) = z + \alpha z^2$, depth $L = 2$, width $N \in [50, 100, 200]$.

E. High dimensionality of input data justifies Gaussian description of fully connected deep networks

We study the CIFAR-10 data set [29], consisting of 10 classes of 32×32 images with 3 color channels. Compared to MNIST, we expect two antagonistic effects. On the one hand, since images within one class of CIFAR-10 are significantly more heterogeneous, we expect the class-conditional distributions to be more complex, and consequently to require higher-order cumulants to accurately represent its statistical structure. On the other hand, due to the larger input dimensionality $N_0 = 3072$ compared to $N_0 = 784$ for MNIST, higher-order cumulants are more strongly suppressed in the input layer (see Sec. III B). To check how these two effects interplay in feed-forward networks, we employ the methods presented in previous sections to restrict training to certain cumulants, similar as in Sec. IV C.

We train network models on the CIFAR-10 data set and compare these to the statistical model trained on the Gaussian approximation of CIFAR-10 (Fig. 7). In both cases, performance is evaluated on the CIFAR-10 test data set. We find that network models trained on data samples achieve performance values of $P = 34.8\% \pm 1.4\%$. In contrast to MNIST, the statistical model trained on the Gaussian statistics consistently achieves *higher* performance values of $P = 37.6\% \pm 1.3\%$. These results are directly linked to the two aforementioned effects: They indicate that due to the large input dimensionality, networks predominantly process only the Gaussian statistics; the statistical model therefore continues to provide a good representation of the network. Moreover, estimates of the Gaussian statistics are more accurate in the statistical model (averaged over the full training set of 50 000 images) compared to training on data samples (averaged over mini-batches of 100 images), possibly explaining the slightly higher performance values. Importantly, although the achieved performance values are far below values reported for other architectures such as convolutional ResNets [30], they are representative for fully connected feed-forward networks [12]. The difference between the architectures lies in the extracted statistical information. For high-dimensional input data, the here presented theory predicts that fully connected feed-forward networks are limited to Gaussian statistics, which can only partly capture the statistical structure of more complex data sets such as CIFAR-10. Hence, the presented decomposition of a network in terms of cumulants allows us to relate the power of network architectures to the processing of statistical information contained in the data.

V. DISCUSSION

The question of how neural networks process data is fundamentally the question of how information is encoded in the data distribution and subsequently transformed by the network. We here present an analytical approach, based on methods from statistical physics, to study the mapping of data distributions implemented by deep feed-forward neural networks: we parametrize the data distribution in terms of correlation functions and derive their successive transformations across layers. We show that the initial network layer effectuates the extraction of information from higher-order correlations in the data; for subsequent layers, a restriction to first- and second-order correlation functions (mean and covariance) already captures the main properties of the network computation. This reduction of the bulk of the network to a nonlinear mapping of a few correlation functions provides an attractive view for further analyses. It relies on the assumption of sufficiently wide layers to apply the central limit theorem, but, in practice, we find that the approximations are useful even for narrow networks.

We validate these results for different data sets. We first investigate an adaptation of the XOR problem that is purely based on first- and second-order cumulants. Despite the nonlinear transformations in each layer giving rise to higher-order correlations, the network solutions to this task can largely be described in terms of transformations solely between mean and covariance of each class. We then consider the MNIST database: we show that network solutions based on empirical estimates for mean and covariance of each class capture a large amount of the variability within the data set, but still exhibit a non-negligible performance gap in comparison to solutions based on the actual data set. We discuss how this performance difference results from the omission of higher-order correlations. We then introduce an example task where higher-order correlations exclusively encode class membership, which allows us to explore their role in isolation. Finally, we show that for high-dimensional input data such as CIFAR-10, the first layer of fully connected networks predominantly extracts the Gaussian statistics. As a consequence, the information processing in these networks is well described by the Gaussian theory.

Limitations. The dimensionality N_0 of the data may limit the applicability of the presented approach to low orders n since cumulants of order n are tensors with N_0^n entries. We note, however, that there exist methods to ease the computational cost of higher-order cumulants in large dimensions: for example, one can make use of the inherent symmetries in these tensors, as well as in the theory itself. The application of such methods to our framework remains a point for future work. A parametrization of a probability distribution in terms of cumulants, moreover, needs to be chosen such that it maintains positivity of the probability density function.

Conserving this property implies constraints for truncating cumulant orders, which require further investigations.

The presented framework and its perturbative methods naturally apply to polynomial approximations of activation functions. Although networks with polynomial nonlinearity are, in principle, not capable of universal function approximation [31–33], this is not an issue for the classification tasks we consider. To obtain illustrative analytical expressions for the mixing of correlation functions, we chose to demonstrate the approach with a quadratic activation function. Nonpolynomial and even nondifferentiable activation functions can, however, also be dealt with in our framework using Gram-Charlier expansions that are detailed for the example of the ReLU activation in Appendix C. While we here mostly focus on the mean and covariance, we also show how to generalize the results to higher-order cumulants.

Relation to kernel limit of deep networks. In this paper we study *individual* networks with specific parameters θ. There is a complementary approach that studies *ensembles* of (infinitely) wide networks with random parameters: Poole *et al.* [15] expose a relation between the Lyapunov exponents and the depth to which information propagates in randomly initialized deep networks. They find the regime close to chaos beneficial for information propagation. We similarly find that the depth scale of information propagation controls the propagation of the Gaussian statistics across data samples studied in the current work, if network parameters are drawn randomly (see Appendix G, Fig. 10). Furthermore, random network parameters are central to studying training as Bayesian inference [34]: independent Gaussian priors on the network parameters render Bayesian inference exact on the resulting Gaussian process [7,12,35,36]. The works [9,37–39] use methods similar to ours to compute finite-width corrections and corrections arising from training with stochastic gradient descent. These approaches consider distributions over network parameters θ. The statistics of the data in this view enters in the form of the pairwise overlaps $\sum_i x_i x'_i$ between pairs of patterns x and x'. In the large data limit, the data statistics can moreover be described by a density $p(x)$, whose properties shape the eigenfunctions ϕ_i of the kernel k in the form $\int k(x, x')\,\phi_i(x')\,p(x')\,dx' = \lambda_i\,\phi_i(x)$ [14], Sec. 4.3. In contrast, in this work we study the transformation of an input distribution $p(x)$ by a network with fixed parameters θ. The focus on individual networks rather than ensembles allows us to directly take into account the internal statistical structure of data samples, for example, in the form of the mean $\mu_{x,i}$ and covariances $\Sigma_{x,ij}$ for individual pixels i and j in images.

Related works. Describing data and network activity in terms of correlations was initially explored by Deco and Brauer [40] on the particular architecture of volume-preserving networks. They derived expressions of the output in terms of its correlations as well as training rules that aim to decorrelate given input data. The work we present here differs in that our goal is not to impose a specific statistical structure on the network output, but to relate the correlations of the input and output distributions and thereby obtain a description of the information processing within the network.

While we do show that these distributions are not exactly Gaussian, that the networks can utilize higher-order correlations in the hidden layers, and how these contributions could in principle be computed, we focus mostly on self-consistently tracking the distributions in Gaussian approximation. This is because, as we show, this approximation is tractable while staying accurate also for trained networks and capturing the majority of the test accuracy in our examples. That a Gaussian approximation is surprisingly effective has also been argued in a recent line of works using teacher-student models with realistic data structure [41–43]. We also derive conditions under which a Gaussian approximation of the activity in the inner layers of a deep network is consistent in the limit of wide layers: Scaling of weight amplitudes $w_{ij} \propto N^{-\frac{1}{2}}$, weak pairwise correlations $c_{ij} = O(N^{-1})$, as well as an approximate pairwise orthogonal decomposition of the previous layer's covariance matrix by the row vectors of the following layer. Under these conditions we show that cumulants of order higher than two are at most $O(N^{-\frac{1}{2}})$. Our approach is inspired by and analogous to the Gaussian equivalence property proposed by [41]; in particular, we also use as the central argument an expansion of higher-order cumulants caused by weak pairwise correlations. Our result differs, though, by us treating layered networks instead of random feature maps embedding a low-dimensional manifold in [41]. Other works which are based on a Gaussian approximation of the representation in each layer are [44] using general deep networks, [45] focusing on ResNets, and [46] considering the case of GANs. Going beyond random weights, other works study dimensionality reduction and decorrelation in both random deep networks and trained deep belief networks [47], explicitly analyzing the effects of weak correlations among weights [48]. Finally, a pedagogical text focusing on field theory for deep neural networks has recently been published [49].

Outlook. Tracing transformations of data correlations through layers of a neural network allows the investigation of mechanisms for both information encoding and processing; in this manner, it presents a handle towards interpretability of deep networks. The availability of tractable expressions describing the transformations of data correlations within neural networks is therefore an interesting prospect for future work seeking to dissect how networks learn and perform tasks. In this context, the theory we propose assumes data statistics of the input distribution $p(x)$ to be known and exposes how statistical features of the data are transformed to generate the output, with the goal of shedding light onto the networks' functioning principles.

Another natural application of the proposed framework is the identification of essential correlations in the data. In that scenario, we do not need the exact distribution $p(x)$, but only sufficiently accurate estimates of some statistics of x that can be obtained from the training data. By manipulating the information available to the model during training, we expose different information encodings the network can employ to solve the same task. We believe this approach could be used to identify data statistics required to solve a given task.

More complex data sets, such as CIFAR-10, require richer network architectures than fully connected feed-forward networks to achieve high performance. For example, applying the presented approach to ResNet-50 [50] would require the extension to convolutional network layers and skip

connections. However, since these are equivalent to linear layers with weight matrices of a particular shape [13], they can straightforwardly be included in the framework.

Another future direction targets expressivity of deep networks: by reversely tracing the data correlations through the network, from target to data, one may ask which input distributions are mapped to a given output distribution, in effect constructing layer-resolved, statistical receptive fields for each target. Expressing these receptive fields in terms of data correlations may also be useful for studying how the complexity of data distributions is reduced by deep neural networks.

ACKNOWLEDGMENTS

We are grateful to C. Merger and A. Kurth for helpful discussions. We thank P. Bouss for feedback on an earlier version of the manuscript. This work was partly supported by the German Federal Ministry for Education and Research (01IS19077A and 01IS19077B), the Excellence Initiative of the German federal and state governments (ERS PF-JARA-SDS005), and the Helmholtz Association Initiative and Networking Fund under Project No. SO-092 (Advanced Computing Architectures, ACA).

APPENDIX A: HIGHER-ORDER CUMULANTS OF POSTACTIVATIONS CAUSED BY WEAKLY CORRELATED PREACTIVATIONS

We study how weak correlations between preactivations affect higher-order cumulants of the postactivations. Assume preactivations x, y are zero-mean Gaussian distributed and weakly correlated. Let ϕ be a piecewise differentiable activation function. The covariance matrix of x and y be $C = \begin{pmatrix} a & c \\ c & a \end{pmatrix}$. For simplicity, we denote $\langle\circ\rangle := \langle\circ\rangle_{(x,y)\sim\mathcal{N}(0,C)}$. Then by Price's theorem [51–53, Appendix A]

$$\frac{\partial}{\partial c}\langle f(x)g(y)\rangle = \langle f'(x)g'(y)\rangle.$$

This can be used to expand $\langle f(x)g(y)\rangle$ for small c as

$$\begin{aligned}\langle f(x)g(y)\rangle &= \langle f(x)g(y)\rangle_{c=0} + \langle f'(x)g'(y)\rangle_{c=0}\, c + O(c^2)\\ &= \langle f(x)\rangle\langle g(y)\rangle + \langle f'(x)\rangle\langle g'(y)\rangle\, c + O(c^2). \quad \text{(A1)}\end{aligned}$$

This expression corresponds to Eq. (A4) in [41], but Goldt *et al.* use a different approach than Price's theorem. In the expression in [41] one needs to replace $\langle uf(u)\rangle = \langle f'(u)\rangle$), which holds since they assume $\langle u^2\rangle = 1$.

Next, we consider the centered variables

$$\tilde{f}(x) := f(x) - \langle f(x)\rangle$$

and correspondingly for g, one gets

$$\langle \tilde{f}(x)\tilde{g}(y)\rangle = \langle f'(x)\rangle\langle g'(y)\rangle\, c + O(c^2).$$

We may generalize this property to expectation values of more than two functions f, g:

$$F_n(x) := \left\langle \prod_{k=1}^{n} \tilde{f}_k(x_k) \right\rangle. \quad \text{(A2)}$$

By the marginalization property of Gaussian distributions, the joint distribution of any subset of x_i is Gaussian distributed, too, where the covariance matrix is the corresponding sector of the matrix $C_{ij} = \langle\langle x_i x_j\rangle\rangle$. Therefore, for any i, j we define the function

$$\begin{aligned}F_n(x\backslash\{x_i, x_j\}) &= \left\langle \prod_{k=1}^{n} \tilde{f}_k(x_k) \right\rangle_{(x_i,x_j)}\\ &= \langle \tilde{f}_i(x_i)\tilde{f}_j(x_j)\rangle_{(x_i,x_j)} \prod_{k\backslash\{i,j\}} \tilde{f}_k(x_k).\end{aligned}$$

Applying (A1) to the first term yields

$$\begin{aligned}F_n(x\backslash\{x_i, x_j\}) &= \left[c_{ij}\, \langle f_i'(x_i) f_j'(x_j)\rangle_{(x_i,x_j),c_{ij}=0} + O\big(c_{ij}^2\big)\right]\\ &\quad\times \prod_{k\backslash\{i,j\}} \tilde{f}_k(x_k). \quad \text{(A3)}\end{aligned}$$

Now take the expectation also across the remaining variables $x\backslash\{x_i, x_j\}$ with probability $p(x\backslash\{x_i, x_j\})$. We may consider $p(x_1, \ldots, x_N) = p(x_i, x_j|x\backslash\{x_i, x_j\})\, p(x\backslash\{x_i, x_j\})$ and use (A3) for the conditional expectation value over x_i, x_j with regard to $p(x_i, x_j|x\backslash\{x_i, x_j\})$, so that it follows

$$\begin{aligned}&\langle F_n(x\backslash\{x_i, x_j\})\rangle_{x\backslash\{x_i,x_j\}}\\ &\quad= \left\langle \left[c_{ij}\, f_i'(x_i) f_j'(x_j) + O\big(c_{ij}^2\big)\right] \prod_{k\backslash\{i,j\}} \tilde{f}_k(x_k) \right\rangle_{x,c_{ij}=0}.\end{aligned}$$

The pair (i, j) has been chosen arbitrary. The remaining factors $\prod_{k\backslash\{i,j\}} \tilde{f}_k(x_k)$ can now be expanded in a similar manner, where all remaining $k\backslash\{i, j\}$ need to be paired. Any such pairing yields nonzero contributions. Together one therefore has

$$\begin{aligned}\big\langle F_n(x)\big\rangle_x &= \sum_{\sigma\in\Pi} c_{\sigma(1)\sigma(2)}\, \langle f'_{\sigma(1)}(x_{\sigma(1)})\rangle\langle f'_{\sigma(2)}(x_{\sigma(2)})\rangle\\ &\quad\times \ldots c_{\sigma(n-1)\sigma(n)}\, \langle f'_{\sigma(n-1)}(x_{\sigma(n-1)})\rangle\langle f'_{\sigma(n)}(x_{\sigma(n)})\rangle\\ &\quad + O\big(c_{\circ\circ}^{\frac{n}{2}+1}\big), \quad \text{(A4)}\end{aligned}$$

where $\sum_{\sigma\in\Pi}$ sums over all disjoint pairings of indices. (This expression corresponds to A16 in [41], apart from minor typos; the factors b_i seem to be missing, p should be m, and we interpret the upper case of their A16 to be meant as $b_1 \ldots b_m \sum_{\sigma\pi\Pi} m_{\sigma_1\sigma_2} \ldots m_{\sigma_{m-1}\sigma_m}$.) This expression is also consistent with Wick's theorem, to which it needs to reduce in the case of an identity mapping $f(x) = x$.

The expansion (A4) holds for arbitrary n. For any n, the result is correct up to terms of order $O(c^{\frac{n}{2}})$. All cumulants of order $n \geqslant 3$ thus vanish at the given order $O(c^{\frac{n}{2}})$. This can be exemplified on the fourth order (dropping the arguments x for brevity)

$$\begin{aligned}\langle\langle \tilde{f}_1\, \tilde{f}_2\, \tilde{f}_3\, \tilde{f}_4\rangle\rangle &= \langle \tilde{f}_1\, \tilde{f}_2\, \tilde{f}_3\, \tilde{f}_4\rangle\\ &\quad - \langle \tilde{f}_1\, \tilde{f}_2\rangle\, \langle \tilde{f}_3\, \tilde{f}_4\rangle\\ &\quad - \langle \tilde{f}_1\, \tilde{f}_3\rangle\, \langle \tilde{f}_2\, \tilde{f}_4\rangle\\ &\quad - \langle \tilde{f}_1\, \tilde{f}_4\rangle\, \langle \tilde{f}_2\, \tilde{f}_3\rangle. \quad \text{(A5)}\end{aligned}$$

The first line on the right-hand side, according to (A4), is

$$\begin{aligned}\langle \tilde{f}_1\,\tilde{f}_2\,\tilde{f}_3\,\tilde{f}_4\rangle &= c_{12}\,\langle f_1'\rangle\,\langle f_2'\rangle\,c_{34}\,\langle f_3'\rangle\,\langle f_4'\rangle \\ &+ c_{13}\,\langle f_1'\rangle\,\langle f_3'\rangle\,c_{24}\,\langle f_2'\rangle\,\langle f_4'\rangle \\ &+ c_{14}\,\langle f_1'\rangle\,\langle f_4'\rangle\,c_{23}\,\langle f_2'\rangle\,\langle f_3'\rangle + O(c^3).\end{aligned}$$

Expanding each of the three negative terms on the right-hand side of (A5) with help of (A4) yields, for example, for the first of them

$$-\langle \tilde{f}_1\,\tilde{f}_2\rangle\,\langle \tilde{f}_3\,\tilde{f}_4\rangle = -c_{12}\,\langle f_1'\rangle\,\langle f_2'\rangle\,c_{34}\,\langle f_3'\rangle\,\langle f_4'\rangle + O(c^3),$$

which precisely cancels the corresponding term in (A5) at the given accuracy $O(c^3)$. Analogous results hold at any even order (odd orders vanish for the centered variables), so that we find

$$\left\langle\!\!\left\langle \prod_{k=1}^{n} \tilde{f}_k(x_k) \right\rangle\!\!\right\rangle = O\big(c^{\frac{n}{2}+1}\big). \tag{A6}$$

We also need to consider the case that indices in (A5) repeat, for example, $\langle\!\langle \tilde{f}_1\,\tilde{f}_1\,\tilde{f}_2\,\tilde{f}_2\rangle\!\rangle$. In general, assume we have r different indices $j_1, \ldots, j_r$ among the n indices and want to compute the nth cumulant for $n > r$. Within a set of repeated variables correlations are of order $O(1)$ instead of $O(c)$. In the expansion (A4) variables with repeated indices must be treated as a single variable. For the given example, define $g_i := \tilde{f}_i^2, i = 1, 2$, and centered variables $\tilde{g}_i := \tilde{f}_i^2 - \langle \tilde{f}_i^2\rangle$. One then has with (A4)

$$\langle \tilde{g}_1\,\tilde{g}_2\rangle = c_{12}\,\langle g_1'\rangle\,\langle g_2'\rangle + O(c^2), \tag{A7}$$

which is also the second cumulant because the $\tilde{g}$ are centered. We then expand the fourth cumulant with repeated indices analogous to (A5) as

$$\begin{aligned}\langle\!\langle \tilde{f}_1\,\tilde{f}_1\,\tilde{f}_2\,\tilde{f}_2\rangle\!\rangle &= \langle \tilde{f}_1\,\tilde{f}_1\,\tilde{f}_2\,\tilde{f}_2\rangle \\ &- \langle \tilde{f}_1\,\tilde{f}_1\rangle\,\langle \tilde{f}_2\,\tilde{f}_2\rangle \\ &- \langle \tilde{f}_1\,\tilde{f}_2\rangle\,\langle \tilde{f}_1\,\tilde{f}_2\rangle \\ &- \langle \tilde{f}_1\,\tilde{f}_2\rangle\,\langle \tilde{f}_1\,\tilde{f}_2\rangle.\end{aligned} \tag{A8}$$

The fourth moment in the first line, using the definitions of g and $\tilde{g}$ above as well as (A7), is

$$\begin{aligned}\langle \tilde{f}_1\,\tilde{f}_1\,\tilde{f}_2\,\tilde{f}_2\rangle &= \langle g_1\,g_2\rangle \\ &= \langle \tilde{g}_1\,\tilde{g}_2\rangle + \langle g_1\rangle\langle g_2\rangle \\ &\overset{(A7)}{=} c_{12}\,\langle g_1'\rangle\,\langle g_2'\rangle + \langle g_1\rangle\langle g_2\rangle + O(c^2).\end{aligned}$$

Combined with (A8) one has

$$\langle\!\langle \tilde{f}_1\,\tilde{f}_1\,\tilde{f}_2\,\tilde{f}_2\rangle\!\rangle = c_{12}\,\langle g_1'\rangle\,\langle g_2'\rangle + O(c^2),$$

where we dropped all terms of order $O(c^2)$, such as $\langle \tilde{f}_1\,\tilde{f}_2\rangle\,\langle \tilde{f}_1\,\tilde{f}_2\rangle = O(c_{12}^2)$ and used that $\langle \tilde{f}_1\,\tilde{f}_1\rangle\,\langle \tilde{f}_2\,\tilde{f}_2\rangle = \langle g_1\rangle\langle g_2\rangle$.

Now consider that r is odd, such as in

$$\begin{aligned}\langle\!\langle \tilde{f}_1\,\tilde{f}_2\,\tilde{f}_2\,\tilde{f}_3\rangle\!\rangle &= \langle \tilde{f}_1\,\tilde{f}_2\,\tilde{f}_2\,\tilde{f}_3\rangle \\ &- \langle \tilde{f}_1\,\tilde{f}_2\rangle\,\langle \tilde{f}_2\,\tilde{f}_3\rangle \\ &- \langle \tilde{f}_1\,\tilde{f}_2\rangle\,\langle \tilde{f}_2\,\tilde{f}_3\rangle \\ &- \langle \tilde{f}_1\,\tilde{f}_3\rangle\,\langle \tilde{f}_2\,\tilde{f}_2\rangle.\end{aligned} \tag{A9}$$

The fourth moment then is

$$\begin{aligned}\langle \tilde{f}_1\,\tilde{f}_2\,\tilde{f}_2\,\tilde{f}_3\rangle &= \langle \tilde{f}_1\,g_2\,\tilde{f}_3\rangle \\ &= \underbrace{\langle \tilde{f}_1\,\tilde{g}_2\,\tilde{f}_3\rangle}_{O(c^2)} + \langle g_2\rangle\,\langle \tilde{f}_1\tilde{f}_3\rangle \\ &\overset{(A7)}{=} \langle g_2\rangle\,\langle f_1'\rangle\langle f_3'\rangle\,c_{13} + O(c^2).\end{aligned}$$

Applied to (A9), we have

$$\begin{aligned}\langle\!\langle \tilde{f}_1\,\tilde{f}_2\,\tilde{f}_2\,\tilde{f}_3\rangle\!\rangle &= \langle g_2\rangle\,\langle f_1'\rangle\langle f_3'\rangle\,c_{13} \\ &- \langle \tilde{f}_1\,\tilde{f}_3\rangle\,\langle \tilde{f}_2\,\tilde{f}_2\rangle + O(c^2) \\ &= O(c^2),\end{aligned}$$

where the terms $\propto c$ cancel exactly. These two examples show the structure of the expansion: If we have r different indices $j_1, \ldots, j_r$, the nth cumulant for $n > r$ of these variables will be of the order in c that equals the number of pairs to join all different indices. So together (r even or odd) we get

$$\left\langle\!\!\left\langle \prod_{k=1}^{n} \tilde{f}_{j_k}(x_{j_k}) \right\rangle\!\!\right\rangle = O\big(c^{\lceil \frac{r}{2} \rceil}\big). \tag{A10}$$

This expression describes the scaling of the higher-order cumulants of postactivations with weak correlations of the preactivations.

APPENDIX B: WEAKLY CORRELATED GAUSSIAN NETWORK MAPPING

The application of a nonlinear activation function ϕ in each network layer generates higher-order cumulants from Gaussian distributed preactivations, as discussed in Sec. III B. We here derive conditions under which higher-order cumulants $G_{z^l}^{(n)}$ of the preactivations z^l beyond mean and covariance on expectation scale down with the layer width N. Consequently, these become negligible for wide networks where $N \gg 1$.

We apply the considerations in the previous Appendix A of weakly correlated Gaussian variables to the network mapping. Preactivations in layer l are given by

$$z_i^l = \sum_{a=1}^{N} W_{ia}^l\, y_a^{l-1} + b_i^l, \tag{B1}$$

which then produce postactivations

$$y_i^l = \phi\big(z_i^l\big). \tag{B2}$$

Assume the preactivations z_i^l are Gaussian and weakly correlated to order ϵ

$$\langle\!\langle z_i z_j\rangle\!\rangle \overset{i\neq j}{=} O(\epsilon). \tag{B3}$$

We want to derive conditions under which it then follows that also preactivations z_i^{l+1} in the next layer have this property. By induction through the layer index, one then has established a condition under which the neglect of non-Gaussian cumulants in the inner layers of the network is justified. To this end, we define centered variables

$$\tilde{z} := z - \langle z \rangle$$

as well as

$$y = f(\tilde{z}) := \phi(\langle z \rangle + \tilde{z}),$$
$$\tilde{y} = \tilde{f}(\tilde{z}) := f(\tilde{z}) - \langle f(\tilde{z}) \rangle.$$

The variance of preactivations should be of order unity

$$\left\langle \left(\tilde{z}_a^l\right)^2 \right\rangle = O(1),$$

because one aims to explore the dynamic range of the gain function, which we assume to be of order unity (we use the dynamic range of the gain function to define our scale). It then follows that also the postactivations have a variance of order unity, so

$$\left\langle \left(\tilde{f}_a^l\right) \right\rangle = O(1).$$

Such conditions are typically also enforced by batch normalization. If the y_a^l were uncorrelated, the variance of the preactivations in the next layer is given by

$$O(1) \overset{!}{=} \left\langle \left(\tilde{z}_i^{l+1}\right)^2 \right\rangle = \sum_a \left[W_{ia}^{l+1}\right]^2 \left\langle \left(\tilde{f}_a^l\right)^2 \right\rangle.$$

For the variances on both sides to be of order unity, we need that

$$W_{ia}^{l+1} = O(N^{-\frac{1}{2}}), \tag{B4}$$

which means that rows and columns of the matrix W^{l+1} are vectors with lengths of order unity.

Now assume the presence of correlations of order

$$\left\langle \tilde{y}_a^l \tilde{y}_b^l \right\rangle = \left\langle \tilde{f}_a^l \tilde{f}_b^l \right\rangle =: C_{ab} = O(\epsilon) \tag{B5}$$

between the outputs of layer l across different neurons $a \neq b$. The expression for the variance then changes to

$$\left\langle \left(\tilde{z}_i^{l+1}\right)^2 \right\rangle = \sum_{a,b} W_{ia}^{l+1} W_{ib}^{l+1} C_{ab}. \tag{B6}$$

To have low correlations in the next layer, one needs to demand that

$$O(\epsilon) \overset{!}{=} \left\langle \tilde{z}_i^{l+1}\, \tilde{z}_j^{l+1} \right\rangle = \sum_{a,b} W_{ia}^{l+1} W_{jb}^{l+1} C_{ab} \quad \forall\, i \neq j. \tag{B7}$$

This can be interpreted as demanding that different rows $W_{i\circ}^{l+1}$ and $W_{j\circ}^{l+1}$ project out mutually nearly orthogonal subspaces out of the space of principal components of C. This means that different neurons i and j each specialize on subspaces that have little mutual overlap.

Now consider higher-order correlations. It follows from (A6) and from the condition of weak pairwise correlations (B3) that for $n \geqslant 3$

$$\left\langle\!\left\langle \prod_{i=1}^n y_i^l \right\rangle\!\right\rangle = \left\langle\!\left\langle \prod_{i=1}^n \tilde{f}_i(\tilde{z}_i^l) \right\rangle\!\right\rangle = O\left(\epsilon^{\frac{n}{2}+1}\right). \tag{B8}$$

The cumulants of the preactivations z_i^{l+1} are given by those of the postactivations as

$$\left\langle\!\left\langle z_{i_1}^{l+1} \dots z_{i_n}^{l+1} \right\rangle\!\right\rangle = \sum_{j_1,\dots,j_n=1}^N W_{i_1 j_1}^{l+1} \dots W_{i_n j_n}^{l+1} \left\langle\!\left\langle \prod_{k=1}^n \tilde{f}_{j_k}(\tilde{z}_{j_k}^l) \right\rangle\!\right\rangle. \tag{B9}$$

We now distinguish three cases:

(1) *Diagonal contributions*. First consider the special case where all indices $j_1 = \dots = j_n$ are identical. One then gets a contribution to (B9) at order $n \geqslant 3$:

$$\begin{aligned} &\sum_{j=1}^N W_{i_1 j}^{l+1} \dots W_{i_n j}^{l+1} \underbrace{\left\langle\!\left\langle \left(\tilde{f}_j\right)^n \right\rangle\!\right\rangle}_{O(1)} \\ &= O\left(N^{1-\frac{n}{2}}\right) \\ &\overset{n\geqslant 3}{<} O\left(N^{-\frac{1}{2}}\right). \end{aligned} \tag{B10}$$

For $n \geqslant 3$ this is suppressed by a large layer width N.

(2) *Off-diagonal contributions with all distinct indices*. Next consider the off-diagonal terms, where all sending neurons' indices are unequal $j_1 \neq j_2 \neq \dots \neq j_n$, so that we can use (B8). For n odd, the contributions vanish because then (B8) vanishes. For n even, we get

$$\begin{aligned} &\sum_{(j_1\neq j_2,\dots,\neq j_n)=1}^N W_{i_1 j_1}^{l+1} \dots W_{i_n j_n}^{l+1} \left\langle\!\left\langle \prod_{k=1}^n \tilde{f}_{j_k}\left(\tilde{z}_{j_k}^l\right) \right\rangle\!\right\rangle \\ &= O\left(\frac{N!}{(N-n)!} N^{-\frac{n}{2}} \epsilon^{\frac{n}{2}+1} \right) \\ &\overset{N\gg n}{=} O\left(N^{\frac{n}{2}} \epsilon^{\frac{n}{2}+1}\right). \end{aligned} \tag{B11}$$

For these contributions to be suppressed for $n \geqslant 3$ with increasing network size, we thus need to demand that the order of pairwise correlations ϵ is at most

$$\epsilon = O(N^{-1}),$$

so that the off-diagonal contribution (B11) is

$$O(N^{-1}),$$

which is hence suppressed with network size also for large orders n.

(3) *Off-diagonal contributions with two or more equal indices*. Now consider terms for which a subset of $j_a, j_b, j_c, \dots$ assume the same value. Let the number of disjoint indices $j_1 \neq j_2 \neq \dots \neq j_r$ be $r < n$. Each pair of equal indices can be seen as the appearance of one Kronecker $\delta_{j_a j_b}$, which eliminates one summation $\sum_{j=1}^N$, hence one factor N less. But at the same time, by (A8), also the moments are increased $\langle \prod_{k=1}^n \tilde{f}_{j_k}(\tilde{z}_{j_k}^l) \rangle = O(\epsilon^{\lceil \frac{r}{2} \rceil})$. Together, we get a contribution

$$\begin{aligned} &O\left(\frac{N!}{(N-r)!} N^{-\frac{n}{2}} \epsilon^{\lceil \frac{r}{2} \rceil} \right) \\ &\overset{N\gg 1,\, \epsilon = O(N^{-1})}{=} O\left(N^r N^{-\frac{n}{2}} N^{-\lceil \frac{r}{2} \rceil}\right) \\ &= O\left(N^{\lfloor \frac{r}{2} \rfloor - \frac{n}{2}}\right) \\ &< O\left(N^{-\frac{1}{2}}\right), \end{aligned} \tag{B12}$$

TABLE II. Interaction functions for different nonlinearities ϕ. We assume $z^l \sim \mathcal{N}(\mu_{z^l}, \Sigma_{z^l})$ in both examples. For ReLU, $\tilde{\mu}_{z^l}$ and $\tilde{\Sigma}_{z^l}$ denote the marginalized mean and covariance with respect to $\tilde{z}^l = (z_i^l, z_j^l)^\mathsf{T}$, and $F_{\tilde{\mu}_{z^l}, \tilde{\Sigma}_{z^l}}(x, y)$ denotes the corresponding cumulative distribution function.

Nonlinearity	Interaction function
$\phi(z) = \mathrm{ReLU}(z)$	$f_{\mu,i} = \frac{\sqrt{\Sigma_{z^l,ii}}}{\sqrt{2\pi}} \exp\left(-\frac{\mu_{z^l,i}^2}{2\Sigma_{z^l,ii}}\right) + \frac{\mu_{z^l,i}}{2}\left[1 + \mathrm{erf}\left(\frac{\mu_{z^l,i}}{\sqrt{2\Sigma_{z^l,ii}}}\right)\right]$
	$f_{\Sigma,ii} = \frac{\Sigma_{z^l,ii}}{2}\left[1 + \mathrm{erf}\left(\frac{\mu_{z^l,i}}{\sqrt{2\Sigma_{z^l,ii}}}\right)\right] + \frac{\mu_{z^l,i}^2}{4} - \left[\frac{\sqrt{\Sigma_{z^l,ii}}}{\sqrt{2\pi}} \exp\left(-\frac{\mu_{z^l,i}^2}{2\Sigma_{z^l,ii}}\right) + \frac{\mu_{z^l,i}}{2}\mathrm{erf}\left(\frac{\mu_{z^l,i}}{\sqrt{2\Sigma_{z^l,ii}}}\right)\right]^2$
	$f_{\Sigma,ij} = \frac{\sqrt{\det(\tilde{\Sigma}_{z^l})}}{2\pi} \exp\left(-\frac{1}{2}\tilde{\mu}_{z^l}^\mathsf{T}\tilde{\Sigma}_{z^l}^{-1}\tilde{\mu}_{z^l}\right)$
	$+\frac{\sqrt{\det(\tilde{\Sigma}_{z^l})}}{2\pi}\mu_{z^l,j}\frac{\sqrt{\pi\,\tilde{\Sigma}^{-1}_{z^l,jj}}}{\sqrt{2}}\exp(-\frac{1}{2}\tilde{\Sigma}^{-1}_{z^l,ii}\mu^2_{z^l,i})\exp\left(\frac{(\tilde{\Sigma}^{-1}_{z^l,ji}\mu_{z^l,i})^2}{2\tilde{\Sigma}^{-1}_{z^l,jj}}\right)\left[1+\mathrm{erf}\left(\frac{(\tilde{\Sigma}^{-1}_{z^l}\tilde{\mu}_{z^l})_j}{\sqrt{2\tilde{\Sigma}^{-1}_{z^l,jj}}}\right)\right]$
	$+\frac{\sqrt{\det(\tilde{\Sigma}_{z^l})}}{2\pi}\mu_{z^l,i}\frac{\sqrt{\pi\,\tilde{\Sigma}^{-1}_{z^l,ii}}}{\sqrt{2}}\exp(-\frac{1}{2}\tilde{\Sigma}^{-1}_{z^l,jj}\mu^2_{z^l,j})\exp\left(\frac{(\tilde{\Sigma}^{-1}_{z^l,ij}\mu_{z^l,j})^2}{2\tilde{\Sigma}^{-1}_{z^l,ii}}\right)\left[1+\mathrm{erf}\left(\frac{(\tilde{\Sigma}^{-1}_{z^l}\tilde{\mu}_{z^l})_i}{\sqrt{2\tilde{\Sigma}^{-1}_{z^l,ii}}}\right)\right]$
	$+\left[\mu_{z^l,i}\mu_{z^l,j} - \tilde{\Sigma}^{-1}_{z^l,ij}\det(\tilde{\Sigma}_{z^l})\right]\left[\frac{1}{2}\mathrm{erf}\left(\frac{\sqrt{2}\mu_{z^l,i}}{\sqrt{\Sigma_{z^l,ii}}}\right) + \frac{1}{2}\mathrm{erf}\left(\frac{\sqrt{2}\mu_{z^l,j}}{\sqrt{\Sigma_{z^l,jj}}}\right) + F_{\tilde{\mu}_{z^l},\tilde{\Sigma}_{z^l}}(0,0)\right]$
	$-\left[\frac{\sqrt{\Sigma_{z^l,ii}}}{\sqrt{2\pi}}\exp\left(-\frac{\mu^2_{z^l,i}}{2\Sigma_{z^l,ii}}\right) + \frac{\mu_{z^l,i}}{2}\left[1+\mathrm{erf}\left(\frac{\mu_{z^l,i}}{\sqrt{2\Sigma_{z^l,ii}}}\right)\right]\right]$
	$\times\left[\frac{\sqrt{\Sigma_{z^l,jj}}}{\sqrt{2\pi}}\exp\left(-\frac{\mu^2_{z^l,j}}{2\Sigma_{z^l,jj}}\right) + \frac{\mu_{z^l,j}}{2}\left[1+\mathrm{erf}\left(\frac{\mu_{z^l,j}}{\sqrt{2\Sigma_{z^l,jj}}}\right)\right]\right]$
$\phi(z) = z + \alpha z^2$	$f_{\mu,i} = \mu_{z^l,i} + \alpha(\mu_{z^l,i})^2 + \alpha\,\Sigma_{z^l,ii}$
	$f_{\Sigma,ij} = \Sigma_{z^l,ij} + 2\alpha\,\Sigma_{z^l,ij}(\mu_{z^l,i} + \mu_{z^l,j}) + 2\alpha^2(\Sigma_{z^l,ij})^2 + 4\alpha^2\mu_{z^l,i}\Sigma_{z^l,ij}\mu_{z^l,j}$

where we used $r < n$ in the last step and upper bounded the expression by the worst case, in which $n = r + 1$, where n is odd. So, contributions from partial diagonal terms are suppressed with network size, too.

In summary, we have shown that the Gaussian approximation with weak pairwise correlations of order $\epsilon < O(N^{-1})$ is consistently maintained in the limit of wide networks $N \gg 1$ if synaptic amplitudes scale as (B4) and if the rows of the connectivity W^l in each layer l in addition obey the approximate orthonormality condition (B7). From a functional perspective the latter condition makes sense because this condition ensures that the N neurons in each layer are used effectively to represent the entire variability that is present in the previous layer, avoiding redundancy among neurons.

Finally, we note that Eq. (B7) is fulfilled for Gaussian initialized, untrained networks. The network parameters $\theta = \{W^l, b^l\}_{l=1,\dots,L+1}$ are drawn i.i.d. from zero-mean Gaussians $W^l_{rs} \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, \sigma_w^2/N_{l-1})$ and $b^l_r \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, \sigma_b^2)$. This choice of initialization precisely preserves the magnitude of the covariance within the network:

$$\begin{aligned}\left\langle \tilde{z}_i^{l+1}\,\tilde{z}_j^{l+1}\right\rangle &= \sum_{a,b} W_{ia}^{l+1}W_{jb}^{l+1}\,C_{ab}^l \\ &\overset{N\gg 1}{\approx} \left\langle \sum_{a,b} W_{ia}^{l+1}W_{jb}^{l+1}\,C_{ab}^l\right\rangle_w \\ &= \delta_{ij}\,\frac{\sigma_w^2}{N}\sum_a C_{aa}^l = O(\epsilon).\end{aligned}$$

Due to the resulting covariance in the next layer being approximately diagonal, the calculations simplify significantly in this case. The above considerations include conditions also for trained networks where correlations among weights cause correlations between pairs of preactivations (z_i^l, z_j^l).

APPENDIX C: INTERACTION FUNCTIONS FOR DIFFERENT ACTIVATION FUNCTIONS

In Sec. III B, we derived the interaction functions resulting from the nonlinearity ϕ:

$$\begin{aligned} f_\mu(\mu_{z^l}, \Sigma_{z^l}) &= \langle\phi(z^l)\rangle_{z^l}, \\ f_\Sigma(\mu_{z^l}, \Sigma_{z^l}) &= \langle\phi(z^l)\,\phi(z^l)^\mathsf{T}\rangle_{z^l} - \mu_{y^l}\mu_{y^l}^\mathsf{T}.\end{aligned}$$

Table II gives these expressions for the ReLU and quadratic nonlinearities.

1. Derivations for ReLU activations

We here consider networks with the ReLU activation function $\phi(z) = \max(0, z)$. Taking the distribution of preactivations z^l to be Gaussian distributed with mean μ_{z^l} and covariance Σ_{z^l}, the mean postactivations are given by

$$\mu_{y^l,i} = \langle\max\left(0, z_i^l\right)\rangle_{z^l\sim\mathcal{N}(\mu_{z^l},\,\Sigma_{z^l})} \tag{C1}$$

$$= \frac{1}{\sqrt{2\pi\,\Sigma_{z^l,ii}}}\int_0^\infty dz_i^l\,z_i^l\,\exp\left(-\frac{\left(z_i^l - \mu_{z^l,i}\right)^2}{2\Sigma_{z^l,ii}}\right) \tag{C2}$$

$$\begin{aligned}&= -\frac{\sqrt{\Sigma_{z^l,ii}}}{\sqrt{2\pi}}\int_{-\mu_{z^l,i}}^\infty dz_i^l\,\frac{-z_i^l}{\Sigma_{z^l,ii}}\exp\left(-\frac{\left(z_i^l\right)^2}{2\,\Sigma_{z^l,ii}}\right) \\ &\quad + \mu_{z^l,i}\,\frac{1}{\sqrt{2\pi\,\Sigma_{z^l,ii}}}\int_{-\mu_{z^l,i}}^\infty dz_i^l\,\exp\left(-\frac{\left(z_i^l\right)^2}{2\,\Sigma_{z^l,ii}}\right)\end{aligned} \tag{C3}$$

$$= \frac{\sqrt{\Sigma_{z^l,ii}}}{\sqrt{2\pi}} \exp\left(-\frac{\mu_{z^l,i}^2}{2\,\Sigma_{z^l,ii}}\right) + \frac{\mu_{z^l,i}}{2}\left[1+\operatorname{erf}\left(\frac{\mu_{z^l,i}}{\sqrt{2\,\Sigma_{z^l,ii}}}\right)\right]. \tag{C4}$$

For the covariance of postactivations, we distinguish the cases $i=j$ and $i\neq j$, starting with the former by calculating its second moment as

$$\langle \phi(z_i^l)\,\phi(z_i^l)\rangle_{z^l\sim\mathcal{N}(\mu_{z^l},\,\Sigma_{z^l})} = \frac{1}{\sqrt{2\pi\,\Sigma_{z^l,ii}}}\int_0^\infty dz_i^l\,(z_i^l)^2\,\exp\left(-\frac{1}{2\Sigma_{z^l,ii}}(z_i^l-\mu_{z^l,i})^2\right) \tag{C5}$$

$$\begin{aligned} &= -\frac{\sqrt{\Sigma_{z^l,ii}}}{\sqrt{2\pi}}\,\mu_{z^l,i}\,\exp\left(-\frac{\mu_{z^l,i}^2}{2\Sigma_{z^l,ii}}\right) + \frac{\Sigma_{z^l,ii}}{2}\left[1+\operatorname{erf}\left(\frac{\mu_{z^l,i}}{\sqrt{2\,\Sigma_{z^l,ii}}}\right)\right] \\ &\quad + \sqrt{\frac{2}{\pi}}\,\mu_{z^l,i}\sqrt{\Sigma_{z^l,ii}}\,\exp\left(-\frac{\mu_{z^l,i}^2}{2\Sigma_{z^l,ii}}\right) + \frac{\mu_{z^l,i}^2}{2}\left[1+\operatorname{erf}\left(\frac{\mu_{z^l,i}}{\sqrt{2\,\Sigma_{z^l,ii}}}\right)\right] \\ &= \frac{\sqrt{\Sigma_{z^l,ii}}\,\mu_{z^l,i}}{\sqrt{2\pi}}\,\exp\left(-\frac{\mu_{z^l,i}^2}{2\Sigma_{z^l,ii}}\right) + \frac{\Sigma_{z^l,ii}+\mu_{z^l,i}^2}{2}\left[1+\operatorname{erf}\left(\frac{\mu_{z^l,i}}{\sqrt{2\,\Sigma_{z^l,ii}}}\right)\right]. \end{aligned} \tag{C6}$$

Combining with the expression for the mean μ_{y^l} then yields the diagonal terms of the covariance:

$$\begin{aligned} \Sigma_{y^l,ii} &= \langle \phi(z_i^l)\,\phi(z_i^l)\rangle_{z^l\sim\mathcal{N}(\mu_{z^l},\,\Sigma_{z^l})} - (\mu_{y^l,i})^2 \\ &= \frac{\Sigma_{z^l,ii}}{2}\left[1+\operatorname{erf}\left(\frac{\mu_{z^l,i}}{\sqrt{2\Sigma_{z^l,ii}}}\right)\right] + \frac{\mu_{z^l,i}^2}{4} \\ &\quad -\left[\frac{\sqrt{\Sigma_{z^l,ii}}}{\sqrt{2\pi}}\,\exp\left(-\frac{1}{2\Sigma_{z^l,ii}}\mu_{z^l,i}^2\right) + \frac{\mu_{z^l,i}}{2}\operatorname{erf}\left(\frac{\mu_{z^l,i}}{\sqrt{2\Sigma_{z^l,ii}}}\right)\right]^2. \end{aligned} \tag{C7, C8}$$

In the case $i\neq j$, we look at the joint distribution of (z_i, z_j) and denote the marginalized mean and covariance by $\tilde{\mu}_z = (\mu_{z,i},\,\mu_{z,j})^\intercal$ and $\tilde{\Sigma}_z = \begin{pmatrix}\Sigma_{z,ii} & \Sigma_{z,ij}\\ \Sigma_{z,ji} & \Sigma_{z,jj}\end{pmatrix}$. For the second moment, we obtain

$$\begin{aligned} &\langle \phi(z_i^l)\,\phi(z_j^l)\rangle_{z^l\sim\mathcal{N}(\mu_{z^l},\,\Sigma_{z^l})} \\ &= \frac{1}{\sqrt{(2\pi)^2\,\det(\tilde{\Sigma}_z)}}\int_0^\infty dz_i^l\int_0^\infty dz_j^l\,z_i^l\,z_j^l \\ &\qquad \times\exp\left(-\frac{1}{2}(\tilde{z}^l-\mu_z)^\intercal\tilde{\Sigma}_z^{-1}(\tilde{z}^l-\mu_z)\right) \end{aligned} \tag{C9}$$

$$\begin{aligned} &= \frac{\sqrt{\det(\tilde{\Sigma}_z)}}{2\pi}\exp\left(-\frac{\tilde{\mu}_z^\intercal\tilde{\Sigma}_z^{-1}\tilde{\mu}_z}{2}\right) \\ &\quad + \frac{\sqrt{\det(\tilde{\Sigma}_z)}}{2\pi}\,\tilde{\Sigma}_{z^l,jj}^{-1}\,\mu_{z^l,j}\,\frac{\sqrt{\pi}}{\sqrt{2\tilde{\Sigma}_{z^l,jj}^{-1}}} \\ &\quad \times\exp\left(-\frac{\tilde{\Sigma}_{z^l,ii}^{-1}\mu_{z^l,i}^2}{2}\right)\exp\left(\frac{\left(\tilde{\Sigma}_{z^l,ji}^{-1}\mu_{z^l,i}\right)^2}{2\tilde{\Sigma}_{z^l,jj}^{-1}}\right) \\ &\quad \times\left[1+\operatorname{erf}\left(\frac{\left(\tilde{\Sigma}_{z^l}^{-1}\tilde{\mu}_{z^l}\right)_j}{\sqrt{2\tilde{\Sigma}_{z^l,jj}^{-1}}}\right)\right] \\ &\quad + \frac{\sqrt{\det(\tilde{\Sigma}_{z^l})}}{2\pi}\,\tilde{\Sigma}_{z^l,ii}^{-1}\,\mu_{z^l,i}\,\frac{\sqrt{\pi}}{\sqrt{2\tilde{\Sigma}_{z^l,ii}^{-1}}} \\ &\quad \times\exp\left(-\frac{\tilde{\Sigma}_{z^l,jj}^{-1}\mu_{z^l,j}^2}{2}\right)\exp\left(\frac{\left(\tilde{\Sigma}_{z^l,ij}^{-1}\mu_{z^l,j}\right)^2}{2\tilde{\Sigma}_{z^l,ii}^{-1}}\right) \\ &\quad \times\left[1+\operatorname{erf}\left(\frac{\left(\tilde{\Sigma}_{z^l}^{-1}\tilde{\mu}_{z^l}\right)_i}{\sqrt{2\tilde{\Sigma}_{z^l,ii}^{-1}}}\right)\right] \\ &\quad + \left(\mu_{z^l,i}\,\mu_{z^l,j} - \tilde{\Sigma}_{z^l,ij}^{-1}\,\det\left(\tilde{\Sigma}_{z^l}\right)\right) \\ &\qquad \times\left[\frac{1}{2}\operatorname{erf}\left(\frac{\sqrt{2}\,\mu_{z^l,i}}{\sqrt{\Sigma_{z^l,ii}}}\right) + \frac{1}{2}\operatorname{erf}\left(\frac{\sqrt{2}\,\mu_{z^l,j}}{\sqrt{\Sigma_{z^l,jj}}}\right) + F_{\tilde{\mu}_{z^l},\tilde{\Sigma}_{z^l}}(0,0)\right], \end{aligned} \tag{C10}$$

where $\tilde{\mu}_{z^l}$ and $\tilde{\Sigma}_{z^l}$ denote the marginalized mean and covariance with respect to $\tilde{z}^l = (z_i^l, z_j^l)^\intercal$, and $F_{\tilde{\mu}_{z^l},\tilde{\Sigma}_{z^l}}(x,y)$ denotes the corresponding cumulative distribution function. $F_{\tilde{\mu}_{z^l},\tilde{\Sigma}_{z^l}}(0,0)$ is also known as the *quadrant probability*. By subtracting $\mu_{y^l,i}\,\mu_{y^l,j}$, we obtain the expression for the cross covariances given in Table II.

Contributions from higher-order correlations

Using the Gram-Charlier expansion [54] of the probability density function $p_{z_i^l}(z_i^l)$, we can derive approximate expressions for the interaction of higher-order correlations of the preactivations z^l. As an example, we derive contributions to the mean of the postactivations y^l up to linear order in $G^{(3)}_{z^l,(i,i,i)}$ for ReLU activations. The Gram-Charlier expansion up to third order is

$$p_{z_i^l}(z_i^l) \approx \left\{1 + \frac{G^{(3)}_{z^l,(i,i,i)}}{3!\sqrt{\Sigma_{z^l,ii}^3}}\left[\left(\frac{z_i^l-\mu_{z^l,i}}{\sqrt{\Sigma_{z^l,ii}}}\right)^3 - 3\frac{(z_i^l-\mu_{z^l,i})}{\sqrt{\Sigma_{z^l,ii}}}\right]\right\} \times\frac{1}{\sqrt{2\pi\,\Sigma_{z^l,ii}}}\exp\left(-\frac{(z_i^l-\mu_{z^l,i})^2}{2\Sigma_{z^l,ii}}\right).$$

Inserting this into the expression for the mean $\mu_{y^l,i}$ of the postactivations y^l, we get

$$\mu_{y^l,i} = \langle \phi(z_i^l) \rangle_{z^l \sim \mathcal{N}(\mu_{z^l}, \Sigma_{z^l})} \tag{C11}$$

$$= \int_0^\infty dz_i^l \, z_i^l \, p_{z_i^l}(z_i^l) \tag{C12}$$

$$\approx \int_0^\infty dz_i^l \, z_i^l \, \frac{1}{\sqrt{2\pi\,\Sigma_{z^l,ii}}} \exp\left(- \frac{\left(z_i^l - \mu_{z^l,i}\right)^2}{2\Sigma_{z^l,ii}} \right) + \frac{G^{(3)}_{z^l,(i,i,i)}}{3!\sqrt{\Sigma^3_{z^l,ii}}} \int_0^\infty dz_i^l \, z_i^l \times \left[\left(\frac{z_i^l - \mu_{z^l,i}}{\sqrt{\Sigma_{z^l,ii}}} \right)^3 - 3 \frac{\left(z_i^l - \mu_{z^l,i}\right)}{\sqrt{\Sigma_{z^l,ii}}} \right] \times \frac{\exp\left(- \left(z_i^l - \mu_{z^l,i}\right)^2 / 2\Sigma_{z^l,ii} \right)}{\sqrt{2\pi\,\Sigma_{z^l,ii}}} \tag{C13}$$

$$= \frac{\sqrt{\Sigma_{z^l,ii}}\,\mu_{z^l,i}}{\sqrt{2\pi}} \exp\left(- \frac{\mu^2_{z^l,i}}{2\,\Sigma_{z^l,ii}} \right) + \frac{\mu_{z^l,i}}{2}\left[1 + \mathrm{erf}\left(\frac{\mu_{z^l,i}}{\sqrt{2\,\Sigma_{z^l,ii}}} \right) \right] - \frac{G^{(3)}_{z^l,(i,i,i)}}{2\,\Sigma^2_{z^l,ii}} \left(\Sigma^2_{z^l,ii} - 1 \right) \frac{1}{2}\left[1 + \mathrm{erf}\left(\frac{\mu_{z^l,i}}{\sqrt{2\,\Sigma_{z^l,ii}}} \right) \right] + \frac{G^{(3)}_{z^l,(i,i,i)}}{3!\,\Sigma^3_{z^l,ii}} \left(3\,\mu_{z^l,i}\Sigma^2_{z^l,ii} + 2\,\Sigma_{z^l,ii} + \mu^3_{z^l,i} + \mu^2_{z^l,i} - 3 \right) \times \frac{\sqrt{\Sigma_{z^l,ii}}\,\mu_{z^l,i}}{\sqrt{2\pi}} \exp\left(- \frac{\mu^2_{z^l,i}}{2\Sigma_{z^l,ii}} \right). \tag{C14}$$

Alternatively, if one wishes to compute higher-order cumulants of y^l, this can be done by first evaluating the integrals for higher-order moments, analogously to the computations above for the first and second moments. Cumulants can then be obtained via the relations given by Gardiner [55].

2. Derivations for quadratic activations

We here consider networks with a quadratic activation function $\phi(z) = z + \alpha z^2$. For any distribution of preactivations z^l with mean μ_{z^l} and covariance Σ_{z^l}, the mean postactivations are given by

$$\mu_{y^l,i} = \left\langle z_i^l + \alpha \left(z_i^l\right)^2 \right\rangle_{z^l} \tag{C15}$$

$$= \mu_{z^l,i} + \alpha \left(\mu_{z^l,i}\right)^2 + \alpha\,\Sigma_{z^l,ii}. \tag{C16}$$

For the covariance of postactivations, we first calculate the second moment

$$\left\langle \phi(z_i^l)\,\phi(z_j^l) \right\rangle_{z^l} = \left\langle \left[z_i^l + \alpha \left(z_i^l\right)^2 \right] \left[z_j^l + \alpha \left(z_j^l\right)^2 \right] \right\rangle_{z^l} \tag{C17}$$

$$= \Sigma_{z^l,ij} + \mu_{z^l,i}\,\mu_{z^l,j} + \alpha\, M^{(3)}_{z^l,(i,j,j)} + \alpha\, M^{(3)}_{z^l,(j,i,i)} + \alpha^2\, M^{(4)}_{z^l,(i,i,j,j)}, \tag{C18}$$

where $M^{(n)}_{z^l}$ denotes the nth moment of preactivations z^l. Combining the expression (C17) for the mean μ_{y^l} then yields the covariance

$$\Sigma_{y^l,ij} = \left\langle \phi(z_i^l)\,\phi(z_j^l) \right\rangle_{z^l} - \left\langle \phi(z_i^l) \right\rangle_{z^l} \left\langle \phi(z_j^l) \right\rangle_{z^l} \tag{C19}$$

$$= \Sigma_{z^l,ij} + 2\,\alpha\,\Sigma_{z^l,ij}\left(\mu_{z^l,i} + \mu_{z^l,j}\right) + 2\,\alpha^2\left(\Sigma_{z^l,ij}\right)^2 + 4\,\alpha^2\,\mu_{z^l,i}\,\Sigma_{z^l,ij}\,\mu_{z^l,j} + \Sigma_{y^l,ij}|_{n>2}, \tag{C20}$$

where $\Sigma_{y^l,ij}|_{n>2}$ contains all terms involving cumulants of order $n > 2$. It is given by

$$\Sigma_{y^l,ij}|_{n>2} = \alpha\left(1 + 2\alpha\,\mu_{z^l,i}\right) G^{(3)}_{z^l,(i,j,j)} + \alpha\left(1 + 2\alpha\,\mu_{z^l,j}\right) G^{(3)}_{z^l,(j,i,i)} + \alpha^2\, G^{(4)}_{z^l,(i,i,j,j)}. \tag{C21}$$

In these expressions, $G^{(n)}_{z^l,(i_1,i_2,\ldots,i_n)}$ denotes the nth cumulant of preactivations given by $\langle\langle z^l_{i_1} z^l_{i_2} \ldots z^l_{i_n} \rangle\rangle$. If the preactivations z^l are Gaussian distributed $z^l \sim \mathcal{N}(\mu_{z^l}, \Sigma_{z^l})$, all cumulants beyond second order vanish, $G^{(n>2)}_{z^l} = 0$, yielding $\Sigma_{y^l,ij}|_{n>2} = 0$ and consequently the result in Table II.

APPENDIX D: INFORMATION PROPAGATION IN NETWORKS WITH QUADRATIC ACTIVATIONS

Figure 2 in the main text illustrates information propagation in networks with ReLU activations. For completeness, we include here as supplemental Fig. 8 the analogous illustration for a network with quadratic activations. For Fig. 3, we include the results for networks with quadratic activation function in supplemental Fig. 9.

APPENDIX E: DATA SAMPLE GENERATION FOR MNIST BASED ON GAUSSIAN APPROXIMATION OF INPUT DISTRIBUTION

In Sec. IV C of the main text, we discuss training networks to solve MNIST using $R_{\mathrm{emp,MSE}}$ [Eq. (25)] with data samples drawn from the Gaussian approximation of the input distribution. For this Gaussian approximation, means $\hat{\mu}^t_x$ and covariances $\hat{\Sigma}^t_x$ for each class t are estimated empirically from the training data set where we flattened the 28×28 images into 784-dimensional vectors. Due to lack of variability in some pixel values at the image edges, the resulting covariances $\hat{\Sigma}^t_x$ are not positive definite, but only positive semidefinite.

To account for the zero eigenvalues of the covariance, data samples are generated based on a principal component analysis of the covariance matrix. For each class t, we decompose the covariance matrix as

$$\hat{\Sigma}^t_x = V D V^\mathsf{T} \tag{E1}$$

with $V = (v_1 | \ldots | v_{N_0})$ containing the unit-length eigenvectors v_i and $D = \mathrm{diag}(\lambda_1, \ldots, \lambda_{N_0})$ containing the corresponding eigenvalues λ_i of $\hat{\Sigma}^t_x$, which we assume to be ordered according to their size, $\lambda_1 \geqslant \cdots \geqslant \lambda_{N_0} \geqslant 0$. We set a threshold $\vartheta_{\mathrm{PCA}} > 0$ that defines a subspace U spanned by the eigenvectors $\{v_i\}_{i=1,\ldots,N_{\mathrm{PCA}}}$ for which $\lambda_i > \vartheta_{\mathrm{PCA}}$. Data samples $\hat{x}^{(d)}|_U$

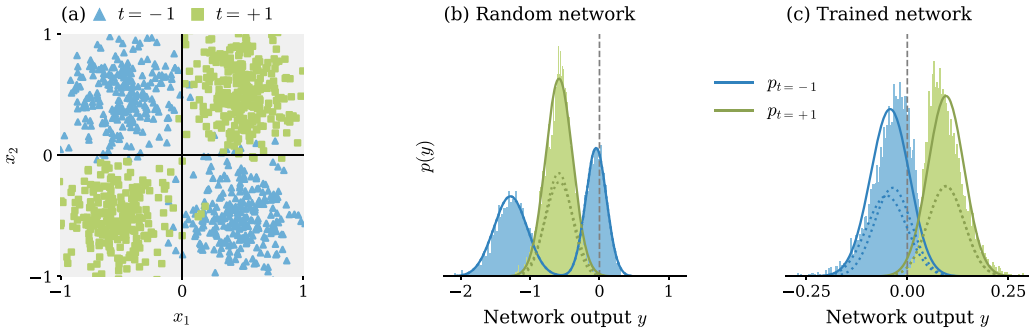


FIG. 8. Information propagation in networks with quadratic activation function for the XOR problem. (a) The distribution of input data is modeled as a Gaussian mixture. Data samples $x^{(d)}$ (blue and green dots) are assigned to class labels $t = \pm 1$ based on the respective mixture component. (b), (c) Distribution of the network output for random (b) and trained (c) parameters. Class-conditional distributions (solid curves) are determined as a superposition of the propagated mixture components (dashed curves) and empirical estimates (blue and green histograms) are obtained from the test data. Since networks are trained on class labels $t = \pm 1$, the classification threshold is set to $y = 0$ (gray lines). Other parameters: $\phi(z) = z + \alpha z^2$, network depth $L = 1$, width $N = 10$; trained network in (c) achieves $P = 90.46\%$ performance.

are then generated with respect to this subspace U and projected back to the input space $\mathbb{R}^{N_0}$ according to

$$\hat{x}^{(d)}|_U \sim \mathcal{N}(0, \mathrm{diag}(\lambda_1, \ldots, \lambda_{N_{\mathrm{PCA}}})), \tag{E2}$$

$$\hat{x}^{(d)} = \hat{\mu}_x^t + V \begin{pmatrix} \hat{x}^{(d)}|_U \\ 0 \end{pmatrix}. \tag{E3}$$

For all experiments in Sec. IV C of the main text, we choose $\vartheta_{\mathrm{PCA}} = 10^{-2}$, corresponding to N_{PCA} between 103 and 234 for the different classes t. Since for all classes t the magnitude of the largest eigenvalue is of order 1, this choice of ϑ_{PCA} ensures including relevant eigenvectors while excluding noise due to finite numerical precision. Since the MNIST training data set contains 60000 samples, to allow for a fair comparison between training on Gaussian samples and on the original images, we generated a similarly sized training data set of $D = 60000$ Gaussian samples.

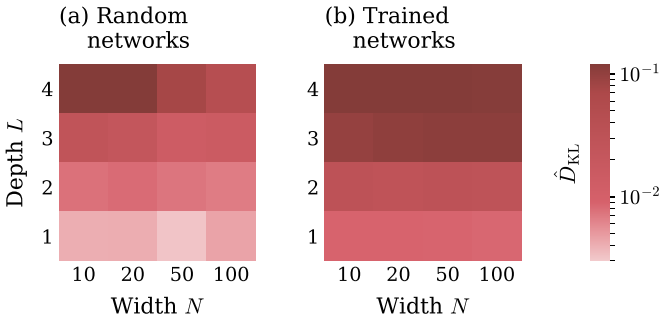


FIG. 9. Deviation between theoretical and empirical output distribution for (a) random and (b) trained networks, measured across 10^2 different network realizations using the normalized Kullback-Leibler divergence $\hat{D}_{\mathrm{KL}}(p_{\mathrm{emp}} \| p_{\mathrm{theo}})$. On average, the trained networks achieve performance values of $P = 96.52\% \pm 0.15\%$ compared to $P_{\mathrm{opt}} = 97.5\%$. Networks were trained to perform the XOR task described in Sec. IV B 1. Other parameters: $\phi(z) = z + \alpha z^2$.

APPENDIX F: PROBLEM SETUP FOR INCLUSION OF HIGHER-ORDER STATISTICS IN THE MAIN TEXT

The problem studied in Sec. IV D of the main text is constructed as follows. We define two classes, $t = \pm 1$, each composed of two Gaussian components $+$ and $-$, with the following means:

$$\mu_x^{t=+1,-} = (-0.5, 0)^\intercal, \quad \mu_x^{t=-1,-} = (-1.5, 0)^\intercal; \tag{F1}$$

$$\mu_x^{t=+1,+} = (1.5, 0)^\intercal, \quad \mu_x^{t=-1,+} = (0.5, 0)^\intercal. \tag{F2}$$

Covariances are isotropic throughout with

$$\Sigma_x^{t,\pm} = 0.05\,\mathbb{I}. \tag{F3}$$

The outer components $(t = -1, -)$ and $(t = +1, +)$ are weighed by $p_{\mathrm{outer}} = \frac{1}{8}$, while the inner components $(t = -1, +)$ and $(t = +1, -)$ are weighed by $p_{\mathrm{inner}} = \frac{3}{8}$, as illustrated in Figs. 6(a) and 6(b) of the main text. A data sample $x^{(d)}$ is assigned a target label $t^{(d)} \in \{\pm 1\}$ based on the mixture component it is drawn from. Distribution parameters are chosen such that the class-conditional means and covariances of the data are identical,

$$\mu_x^{t=\pm 1} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \quad \Sigma_x^{t=\pm 1} = \begin{pmatrix} 0.8 & 0 \\ 0 & 0.5 \end{pmatrix}, \tag{F4}$$

while the third-order correlations differ

$$G^{(3),\,t=\pm 1}_{x,\,(i,j,k)} = \pm 0.75\,\delta_{ij}\delta_{jk}\delta_{ki}\delta_{i1}. \tag{F5}$$

We use training and test data sets of size $D = 10^4$.

APPENDIX G: DEPTH SCALES OF INFORMATION PROPAGATION

We here discuss the relation between the presented work and Poole *et al.* [15]. To formalize this relation, we define as $z_{\theta k}^{ld}$ the preactivation of neuron k in layer l for a given data sample $x^{(d)}$ in a network with parameters θ. Poole *et al.* study ensembles of networks across random realizations of network parameters θ. The family of distributions they study,

expressed in terms of preactivations, is thus

$$\tilde{p}^{l}_{\{d\}}(\{z_k\}) = \left\langle \prod_{k,d} \delta\left(z_k^d - z_{k\theta}^{ld}\right) \right\rangle_{\theta}, \tag{G1}$$

which is one distribution jointly for all preactivations $\{z_k^d\}_{d=1,\dots,D;\,k=1,\dots,N}$ for a given set of P data samples $x^{(d)}$ and for each given layer l. In the limit of wide networks, they find that $\tilde{p}$ factorizes across different neuron indices, so that z_i^{ld} and $z_k^{ld'}$ are independent for different $i \neq k$. Further, these variables are centered Gaussian, so that a single covariance matrix is sufficient to describe their statistics. In this limit, it is therefore sufficient to study the joint statistics of all pairs of networks corresponding to all pairs of inputs d, d':

$$\tilde{p}^{l}_{dd'}(z, z') = \left\langle \delta\left(z - z_{\theta}^{ld}\right) \delta\left(z' - z_{\theta}^{ld'}\right) \right\rangle_{\theta}. \tag{G2}$$

Correlation functions in their work *a priori* thus quantify fluctuations across realizations of network parameters. Their mean-field theory for deep feed-forward networks is identical to the classical mean-field theory of random recurrent networks [56] because for recurrent networks with discrete-time updates the equal-time statistics is identical to the equal-layer statistics of a deep network [57].

In the presented work, instead, we study individual networks defined by one fixed set of parameters θ across the distribution $p(x)$ of data samples $x^{(d)}$. Correlations in our work thus quantify the variability of the network state across different data points. Formally, the family of distributions we study is

$$p^{l}_{\theta}(\{z_k\}) = \left\langle \prod_{k} \delta\left(z_k - z_{\theta}^{ld}\right) \right\rangle_{d}, \tag{G3}$$

which for each given l and θ is one joint distribution of all neurons k. Importantly, the distribution is across the ensemble of data points d.

One formal difference is thus the expectation across θ in (G1) versus the expectation over d in (G3). Wide networks, however, tend to be self-averaging. This means that the ensemble across parameters θ studied by Poole *et al.* shows a concentration on a single typical behavior that one finds in any of its (likely) individual realizations. Formally, this means that the empirical distribution of (z_k, z'_k) across neurons k for any random choice of parameters θ takes on the same form as $\tilde{p}$, so that (G2) for N large approaches the empirical average over neuron activations,

$$\tilde{p}^{l}_{dd'}(z, z') \overset{\text{self-averaging}}{\simeq} N^{-1} \sum_{k} \delta\left(z - z_{\theta k}^{ld}\right) \delta\left(z' - z_{\theta k}^{ld'}\right), \quad \forall \theta.$$

A way to show this is by a saddle-point approximation of the moment-generating function after the disorder average across θ [e.g., 21,53,57–59].

To derive the result by Poole *et al.* [15] or Molgedey *et al.* [56] in our notation, we start with the expression for the preactivations $z_i^{l+1} = \sum_k W_{ik}^{l+1} y_k^l + b_i^{l+1}$ [see Eq. (1)]. For Gaussian distributed W_{ik}^{l+1} and b_i^{l+1}, preactivations for one fixed data sample $x^{(d)}$ (suppressing the superscript d for brevity in the following) become Gaussian as well, with mean and covariance

$$\begin{aligned} M_{z^{l+1},\,i} &:= \left\langle \sum_k W_{ik}^{l+1} y_k^l + b_i^{l+1} \right\rangle_{W,b} \\ &= \sum_k \left\langle W_{ik}^{l+1} \right\rangle_{W^{l+1}} \left\langle y_k^l \right\rangle_{W,b} + \left\langle b_i^{l+1} \right\rangle_{b^{l+1}} = 0, \end{aligned} \tag{G4}$$

$$\begin{aligned} S_{z^{l+1},ij} &:= \left\langle \sum_{k,m} W_{ik}^{l+1} W_{jm}^{l+1} y_k^l y_m^l + b_i^{l+1} b_j^{l+1} \right\rangle_{W,b} \\ &= \sum_{k,m} \left\langle W_{ik}^{l+1} W_{jm}^{l+1} \right\rangle_{W^{l+1}} \left\langle y_k^l y_m^l \right\rangle_{W,b} + \left\langle b_i^{l+1} b_j^{l+1} \right\rangle_{b^{l+1}} \\ &= \delta_{ij} \left(\sigma_w^2 \left\langle y^l y^l \right\rangle_{W,b} + \sigma_b^2 \right) \\ &=: \delta_{ij}\, S_{z^{l+1}}, \end{aligned} \tag{G5}$$

with

$$S_{z^{l+1}} = \sigma_w^2 \left\langle \phi(z^l)\phi(z^l) \right\rangle_{z^{l+1} \sim \mathcal{N}(0, S_{z^l})} + \sigma_b^2, \tag{G6}$$

and where we used the mapping by the activation function (15). To determine the statistics $\langle y_k^l \rangle_{W,b}$ and $\langle y_k^l y_k^l \rangle_{W,b}$, we simultaneously performed an average over weights and biases in layers $l' \leqslant l$. These statistics are identical across neurons, so we write $\langle y_k^l \rangle_{W,b} = \langle y^l \rangle_{W,b}$ and $\langle y_k^l y_k^l \rangle_{W,b} = \langle y^l y^l \rangle_{W,b}$. Equations (G4) and (G5) show that correlations among different neurons vanish on average across networks.

The covariance between preactivations of a pair of networks for two different inputs $x^{(d)}$ and $x^{(d')}$ analogously becomes

$$\begin{aligned} &S_{z^{l+1,d} z^{l+1,d'}} \\ &\quad = \sigma_w^2 \left\langle \phi(z^{l,d})\phi(z^{l,d'}) \right\rangle_{(z^{l,d}, z^{l,d'}) \sim \mathcal{N}(0, \{S_{z^{l,d} z^{l,d'}}\})} + \sigma_b^2, \end{aligned} \tag{G7}$$

where $\mathcal{N}(0, \{S_{z^{l,d} z^{l,d'}}\})$ is meant as the Gaussian distribution for the pair $(z^{l,d}, z^{l,d'})$ with covariance matrix $\begin{pmatrix} S_{z^{l,d}} & S_{z^{l,d} z^{l,d'}} \\ S_{z^{l,d'} z^{l,d}} & S_{z^{l,d'}} \end{pmatrix}$.

We may make a connection to our results by considering a pair of inputs $x^{(d)}$ and $x^{(d')}$. These inputs are presented to the network as y^{0d} and $y^{0d'}$. The theory by Poole *et al.* yields a measure of the overlap $O^l_{dd'}$ of network states after l layers as the solution of the joint iterative equations derived above. In the limit of wide networks, this overlap becomes self-averaging, so it concentrates around its mean value across y:

$$\begin{aligned} O^l_{dd'} &:= N^{-1} \sum_k y_k^{ld} y_k^{ld'} \\ &\simeq \left\langle y^{ld} y^{ld'} \right\rangle_{W,b} \\ &= \left\langle \phi(z^{l,d})\phi(z^{l,d'}) \right\rangle_{(z^{l,d}, z^{l,d'}) \sim \mathcal{N}(0, \{S_{z^{l,d} z^{l,d'}}\})} \\ &= \sigma_w^{-2} \left(S_{z^{l+1,d} z^{l+1,d'}} - \sigma_b^2 \right). \end{aligned} \tag{G8}$$

To show the simplest possible link between the depth scales studied in Poole *et al.*, we consider the case of a deep untrained network. The statistics of z^l and y^l decay to a fixed point, so that we can consider the autostatistics to become

constant for a large enough l,

$$\begin{aligned} S_{z^{l,d}} &= A_0, \quad \forall\, d, \\ S_{z^{l,d}z^{l,d'}} &= C_0, \quad \forall\, d \neq d', \end{aligned} \tag{G9}$$

where A_0 is the stationary solution of (G6),

$$A_0 = \sigma_w^2 \,\langle \phi(z)\phi(z)\rangle_{z\sim\mathcal{N}(0,A_0)} + \sigma_b^2, \tag{G10}$$

and C_0 the stationary solution of (G7),

$$C_0 = \sigma_w^2 \,\langle \phi(z_1)\phi(z_2)\rangle_{(z_1,z_2)\sim\mathcal{N}\left(0,\left[\begin{smallmatrix} A_0 & C_0 \\ C_0 & A_0 \end{smallmatrix}\right]\right)} + \sigma_b^2.$$

Now consider pairs of inputs $(y^{0d}, y^{0d'})$ for which the statistics of preactivations differ only little from this fixed-point statistics: assume that any data point has variance $S_{z^l} = A_0$ and for any pair (d, d') of data points we may express the covariance of preactivations in the first layer as

$$S_{z^{1,d}z^{1,d'}} = C_0 + \delta C^1_{dd'}, \tag{G11}$$

where $\delta C^1_{dd'} \ll C_0$. Based on these assumptions, we now compute decay constants with l.

Linearizing the iteration (G7), one obtains for the propagation of $\delta C^l_{dd'}$

$$\delta C^{l+1}_{dd'} = \sigma_w^2 \,\langle \phi'(z^{l,d})\phi'(z^{l,d'})\rangle\, \delta C^l_{dd'} + O\big[\big(\delta C^l_{dd'}\big)^2\big].$$

Here, we made use of Price's theorem [51–53, Appendix A] $\partial\langle\phi(z)\phi(z')\rangle/\partial\Sigma_{zz'} = \langle\phi'(z)\phi'(z')\rangle$ where $\phi' = d\phi/dz$ and $\Sigma_{zz'}$ is the covariance of z and z'. For stationary statistics across layers (G10) and under the homogeneity assumption across data samples (G9), one thus has

$$\begin{aligned} &\langle \phi'(z^d)\phi'(z^{d'})\rangle \\ &\quad = \langle \phi'(z_1)\phi'(z_2)\rangle_{(z_1,z_2)\sim\mathcal{N}\left(0,\left[\begin{smallmatrix} A_0 & C_0 \\ C_0 & A_0 \end{smallmatrix}\right]\right)} \quad \forall\, d, d' \\ &\quad =: \langle \phi'\phi'\rangle. \end{aligned}$$

One then obtains an exponential evolution with layer index

$$\begin{aligned} \delta C^{l+1}_{dd'} &= \big(\sigma_w^2 \,\langle \phi'\phi'\rangle\big)^l \,\delta C^1_{dd'} \\ &= e^{-\frac{l}{\xi}}\, \delta C^1_{dd'}, \end{aligned} \tag{G12}$$

with a depth scale

$$\xi = -1/\ln\big[\sigma_w^2 \,\langle \phi'\phi'\rangle\big]. \tag{G13}$$

So, this equation gives rise to the depth scales studied in Poole *et al.* for network ensembles. This scale ξ^{-1} corresponds to the Lyapunov exponent computed in Molgedey *et al.* In particular, at the transition to chaos, namely, at the point in parameter space (σ_w, σ_b) for which $\sigma_w^2 \,\langle \phi'\phi'\rangle = 1$, the depth scale diverges. The overlap of activations (G8) shows the same depth scale because its variation is linearly related to $\delta C^l_{dd'}$ as $\delta C^l_{dd'} = \sigma_w^2\, \delta O^{l-1}_{dd'}$, so

$$\delta O^l_{dd'} = \big(\sigma_w^2 \,\langle \phi'\phi'\rangle\big)^l \,\delta O^0_{dd'}. \tag{G14}$$

Both the covariance of preactivations $(S_{z^{1,d}z^{1,d'}})$ and the overlaps of activations $(O^l_{dd'} = \delta O^l_{dd'} + O_0)$ therefore decay to fixed points, respectively C_0 and O_0, that are related by $O_0 = \sigma_w^{-2}(C_0 - \sigma_b^2)$.

We can relate these results to our work on single networks by reexpressing the overlap $O^l_{dd'}$ in terms of the probability distribution across different data samples. From (G8) it follows that

$$\frac{1}{D\,(D-1)} \sum_{(d\neq d')=1}^{D} O^l_{dd'} \tag{G15}$$

$$\simeq\; N^{-1} \sum_{k=1}^{N} \left[\frac{1}{D}\sum_{d=1}^{D} y_k^{ld}\right]\left[\frac{1}{D}\sum_{d'=1}^{D} y_k^{ld'}\right] + O(D^{-1}) \tag{G16}$$

$$\overset{D\gg 1}{\simeq}\; N^{-1} \sum_{k=1}^{N} \big\langle y_k^{ld}\big\rangle_d^2 \tag{G17}$$

$$\simeq\; N^{-1} \sum_{k=1}^{N} (\mu_{y^l,\{k\}})^2. \tag{G18}$$

Here, $\mu_{y^l,\{k\}}$ denotes the mean postactivation of neuron k in layer l, taken over the ensemble of all data points d. This is obtained by iterating Eqs. (14) and (17).

We show in Fig. 10 that predictions of (G18) are indeed consistent with the depth scale obtained from Poole *et al.*'s theory (G13). Moreover, since we derived our theory for single networks, (G18) also captures variability due to particular network realizations. Interestingly, while for network ensembles the depth scale ξ describes the evolution of the second moments, expression (G18) shows that for single networks ξ describes the evolution of the squared means across data samples.

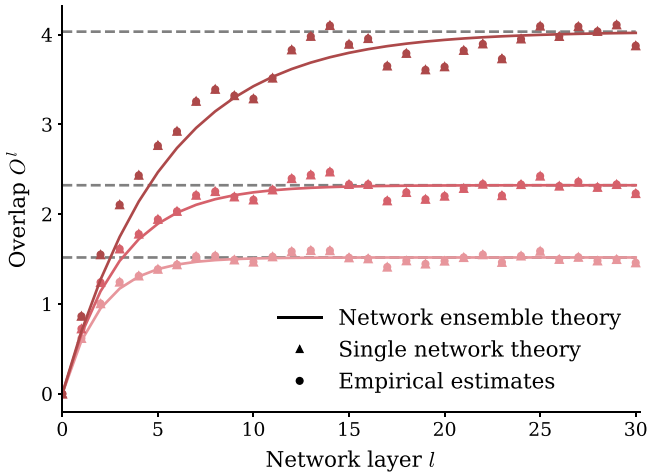


FIG. 10. Depth scale of cumulant propagation in a randomly initialized network. Evolution of overlaps O^l in (G8) and (G18) as a function of the layer l. Solid curves show the predicted decay as $e^{-l/\xi}$, with length scale ξ given by (G13) from Poole *et al.* for network ensembles. Dashed lines indicate the fixed points O_0 to which the overlaps converge. Empirical estimates of the overlaps $O^l_{dd'}$ in (G15) are shown as dots. Values of the overlaps based on the cumulant propagation for single networks derived in (G18) are shown as triangles. Empirical estimates and the values based on data statistics match closely so that the symbols overlap. To isolate the depth scale of the overlaps, input data $\{x^{(d)}\}_d$ are drawn from a Gaussian $\mathcal{N}(0, A_0)$, with A_0 given by (G10). Other parameters: $\sigma_w = \sigma_b \in [0.76, 0.81, 0.85]$ (from light to dark colors), $\alpha = 0.1$.

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, Imagenet classification with deep convolutional neural networks, in *Advances in Neural Information Processing Systems*, edited by F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger (Curran Associates, Red Hook, NY, 2012), Vol. 25, pp. 1097–1105, https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf.

[2] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot *et al.*, Mastering the game of go with deep neural networks and tree search, Nature (London) **529**, 484 (2016).

[3] C. M. Bishop, *Pattern Recognition and Machine Learning* (Springer, New York, 2006).

[4] Y. Bahri, J. Kadmon, J. Pennington, S. S. Schoenholz, J. Sohl-Dickstein, and S. Ganguli, Statistical mechanics of deep learning, Annu. Rev. Condens. Matter Phys. **11**, 501 (2020).

[5] H. W. Lin, M. Tegmark, and D. Rolnick, Why does deep and cheap learning work so well? J. Stat. Phys. **168**, 1223 (2017).

[6] R. Shwartz-Ziv and N. Tishby, Opening the black box of deep neural networks via information, arXiv:1703.00810.

[7] A. Jacot, F. Gabriel, and C. Hongler, Neural tangent kernel: Convergence and generalization in neural networks, in *Advances in Neural Information Processing Systems* (MIT Press, Cambridge, MA, 2018), Vol. 31, pp. 8580–8589, https://proceedings.neurips.cc/paper/2018/file/5a4be1fa34e62bb8a6ec6b91d2462f5a-Paper.pdf.

[8] A. M. Saxe, J. L. McClelland, and S. Ganguli, A mathematical theory of semantic development in deep neural networks, Proc. Natl. Acad. Sci. USA **116**, 11537 (2019).

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

[10] R. M. Neal, *Bayesian Learning for Neural Networks* (Springer, New York, 1996).

[11] C. K. Williams, Computation with infinite neural networks, Neural Comput. **10**, 1203 (1998).

[12] J. Lee, J. Sohl-Dickstein, J. Pennington, R. Novak, S. Schoenholz, and Y. Bahri, Deep neural networks as gaussian processes, in International Conference on Learning Representations (2018), https://openreview.net/forum?id=B1EA-M-0Z.

[13] A. Garriga-Alonso, C. E. Rasmussen, and L. Aitchison, Deep convolutional networks as shallow gaussian processes, in International Conference on Learning Representations (2019), https://openreview.net/forum?id=Bklfsi0cKm.

[14] C. Rasmussen and C. Williams, *Gaussian Processes for Machine Learning, Adaptive Computation and Machine Learning* (MIT Press, Cambridge, MA, 2006), p. 248.

[15] B. Poole, S. Lahiri, M. Raghu, J. Sohl-Dickstein, and S. Ganguli, Exponential expressivity in deep neural networks through transient chaos, in *Advances in Neural Information Processing Systems 29*, edited by D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett (Curran Associates, Red Hook, NY, 2016), pp. 3360–3368, https://proceedings.neurips.cc/paper/2016/file/148510031349642de5ca0c544f31b2ef-Paper.pdf.

[16] M. Raghu, B. Poole, J. Kleinberg, S. Ganguli, and J. Sohl-Dickstein, On the expressive power of deep neural networks, in *Proceedings of the 34th International Conference on Machine Learning, Proceedings of Machine Learning Research*, Vol. 70, edited by D. Precup and Y. W. Teh (PMLR, 2017), pp. 2847–2854, https://proceedings.mlr.press/v70/raghu17a.html.

[17] S. S. Schoenholz, J. Gilmer, S. Ganguli, and J. Sohl-Dickstein, Deep information propagation, arXiv:1611.01232.

[18] H. Kleinert, *Gauge Fields in Condensed Matter, Vol. I, Superflow and Vortex Lines Disorder Fields, Phase Transitions* (World Scientific, Singapore, 1989).

[19] J. Zinn-Justin, *Quantum Field Theory and Critical Phenomena* (Clarendon, Oxford, 1996).

[20] J. A. Hertz, Y. Roudi, and P. Sollich, Path integral methods for the dynamics of stochastic and disordered systems, J. Phys. A: Math. Theor. **50**, 033001 (2017).

[21] M. Helias and D. Dahmen, *Statistical Field Theory for Neural Networks* (Springer, Berlin, 2020), Vol. 970, p. 203.

[22] Y. LeCun, C. Cortes, and C. J. Burges, The mnist database of handwritten digits, 1998, http://yann.lecun.com/exdb/mnist/.

[23] V. Vapnik, Principles of risk minimization for learning theory, in *Advances in Neural Information Processing Systems*, edited by J. Moody, S. Hanson, and R. P. Lippmann (Morgan Kaufmann, San Francisco, 1992), Vol. 4, pp. 831–838, https://proceedings.neurips.cc/paper/1991/file/ff4d5fbbafdf976cfdc032e3bde78de5-Paper.pdf.

[24] V. N. Vapnik, Adaptive and learning systems for signal processing communications, and control, *The Nature of Statistical Learning Theory* (Springer, New York, 1998).

[25] R. Kohavi and D. H. Wolpert, Bias plus variance decomposition for zero-one loss functions, in *Proceedings of the Thirteenth International Conference on Machine Learning* (Morgan Kaufmann, San Francisco, 1996), Vol. 96, pp. 275–283.

[26] D. P. Kingma and J. L. Ba, Adam: A method for stochastic gradient descent, in International Conference on Learning Representations (2015), https://openreview.net/forum?id=8gmWwjFyLj.

[27] I. Loshchilov and F. Hutter, Decoupled weight decay regularization, in International Conference on Learning Representations (2019), https://openreview.net/forum?id=Bkg6RiCqY7.

[28] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai *et al.*, Pytorch: An imperative style, high-performance deep learning library, in *Advances in Neural Information Processing Systems 32*, edited by H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Curran Associates, Red Hook, NY, 2019), Vol. 32, pp. 8024–8035, https://proceedings.neurips.cc/paper/2019/file/bdbca288fee7f92f2bfa9f7012727740-Paper.pdf.

[29] A. Krizhevsky , Learning multiple layers of features from tiny images, Department of Computer Science, University of Toronto, 2009, https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf.

[30] S. Zagoruyko and N. Komodakis, Wide residual networks, in Proceedings of the British Machine Vision Conference (BMVC), edited by E. R. H. Richard C. Wilson and W. A. P. Smith (BMVA Press, Durham, UK, 2016), pp. 87.1–87.12, https://dx.doi.org/10.5244/C.30.87.

[31] G. Cybenko, Approximation by superpositions of a sigmoidal function, Math. Control Signals Syst. **2**, 303 (1989).

[32] M. Leshno, V. Y. Lin, A. Pinkus, and S. Schocken, Multilayer feedforward networks with a nonpolynomial activation func-

tion can approximate any function, Neural Networks **6**, 861 (1993).

[33] A. Pinkus, Approximation theory of the mlp model in neural networks, Acta Numer. **8**, 143 (1999).

[34] D. J. MacKay, *Information Theory, Inference and Learning Algorithms* (Cambridge University Press, Cambridge, 2003).

[35] C. K. I. Williams and D. Barber, Bayesian classification with Gaussian processes, IEEE Trans. Pattern Anal. Mach. Intell. **12**, 1342 (1998).

[36] C. K. Williams and C. E. Rasmussen, *Gaussian Processes for Machine Learning*, 1st ed. (MIT Press, Cambridge, MA, 2006).

[37] E. Dyer and G. Gur-Ari, Asymptotics of wide networks from feynman diagrams, in International Conference on Learning Representations (2020), https://openreview.net/forum?id=S1gFvANKDS.

[38] G. Naveh, O. Ben David, H. Sompolinsky, and Z. Ringel, Predicting the outputs of finite deep neural networks trained with noisy gradients, Phys. Rev. E **104**, 064301 (2021).

[39] S. Yaida, Non-Gaussian processes and neural networks at finite widths, in *Proceedings of The First Mathematical and Scientific Machine Learning Conference*, Proceedings of Machine Learning Research, edited by J. Lu and R. Ward (Princeton University, Princeton, NJ, 2020), Vol. 107, pp. 165–192, http://proceedings.mlr.press/v107/yaida20a/yaida20a.pdf.

[40] G. Deco and W. Brauer, Higher order statistical decorrelation without information loss, in *Proceedings of the 7th International Conference on Neural Information Processing Systems, NIPS'94* (MIT Press, Cambridge, MA, 1994), pp. 247–254, https://proceedings.neurips.cc/paper/1994/file/892c91e0a653ba19df81a90f89d99bcd-Paper.pdf.

[41] S. Goldt, M. Mézard, F. Krzakala, and L. Zdeborová, Modeling the Influence of Data Structure on Learning in Neural Networks: The Hidden Manifold Model, Phys. Rev. X **10**, 041044 (2020).

[42] S. Goldt, G. Reeves, M. Mézard, F. Krzakala, and L. Zdeborová, The Gaussian equivalence of generative models for learning with two-layer neural networks, arXiv:2006.14709.

[43] B. Loureiro, C. Gerbelot, H. Cui, S. Goldt, F. Krzakala, M. Mézard, and L. Zdeborová, Capturing the learning curves of generic features maps for realistic data sets with a teacher-student model, arXiv:2102.08127.

[44] G. Yang and E. J. Hu, Tensor programs iv: Feature learning in infinite-width neural networks, in *Proceedings of the 38th International Conference on Machine Learning*, Proceedings of Machine Learning Research, Vol. 139, edited by M. Meila and T. Zhang (PMLR, 2021), pp. 11727–11737, https://proceedings.mlr.press/v139/yang21c.html.

[45] C. Fang, J. Lee, P. Yang, and T. Zhang, Modeling from features: a mean-field framework for over-parameterized deep neural networks, in *Proceedings of Thirty Fourth Conference on Learning Theory*, Proceedings of Machine Learning Research, Vol. 134 (PMLR, 2021), pp. 1887–1936, https://proceedings.mlr.press/v134/fang21a.html.

[46] M. E. A. Seddik, C. Louart, M. Tamaazousti, and R. Couillet, Random Matrix Theory Proves that Deep Learning Representations of GAN-data Behave as Gaussian Mixtures, in *International Conference on Machine Learning* (PMLR Press, 2020), pp. 8573–8582, http://proceedings.mlr.press/v119/seddik20a.html.

[47] H. Huang, Mechanisms of dimensionality reduction and decorrelation in deep neural networks, Phys. Rev. E **98**, 062313 (2018).

[48] J. Zhou and H. Huang, Weakly correlated synapses promote dimension reduction in deep neural networks, Phys. Rev. E **103**, 012315 (2021).

[49] D. A. Roberts, S. Yaida, and B. Hanin, *The Principles of Deep Learning Theory: An Effective Theory Approach to Understanding Neural Networks* (Cambridge University Press, Cambridge, in press).

[50] K. He, X. Zhang, S. Ren, and J. Sun, Deep residual learning for image recognition, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (IEEE, Piscataway, NJ, 2016), https://ieeexplore.ieee.org/document/7780459.

[51] R. Price, A useful theorem for nonlinear devices having gaussian inputs, IEEE Trans. Inf. Theory **4**, 69 (1958).

[52] A. Papoulis and S. U. Pillai, *Probability, Random Variables, and Stochastic Processes*, 4th ed. (McGraw-Hill, Boston, 2002).

[53] J. Schuecker, S. Goedeke, D. Dahmen, and M. Helias, Functional methods for disordered neural networks, arXiv:1605.06758.

[54] S. Blinnikov and R. Moessner, Expansions for nearly Gaussian distributions, Astron. Astrophys. Suppl. Ser. **130**, 193 (1998).

[55] C. W. Gardiner, *Handbook of Stochastic Methods for Physics, Chemistry and the Natural Sciences*, 2nd ed., Springer Series in Synergetics No. 13 (Springer, Berlin, 1985).

[56] L. Molgedey, J. Schuchhardt, and H. G. Schuster, Suppressing Chaos in Neural Networks by Noise, Phys. Rev. Lett. **69**, 3717 (1992).

[57] K. Segadlo, B. Epping, A. van Meegen, D. Dahmen, M. Krämer, and M. Helias, Unified field theory for deep and recurrent neural networks, J. Stat. Mech. (2022) 103401.

[58] A. Crisanti and H. Sompolinsky, Path integral approach to random neural networks, Phys. Rev. E **98**, 062120 (2018).

[59] B. Bordelon and C. Pehlevan, Self-consistent dynamical field theory of kernel evolution in wide neural networks, arXiv:2205.09653.