Politecnico di Torino

A data-agnostic statistical mechanics approach for studying deep neural networks beyond the infinite-width limit

Rosalba Pacelli

PHD thesis submitted to

Politecnico di Torino

Supervisor

Prof. Riccardo Zecchina

Co-Supervisor

Dr Pietro Rotondo

Milan, October 2023

Abstract

A decade of technological advancement driven by the success of deep learning in various tasks [1–3] is not yet supported by a theoretical framework able to capture the features of architectures, loss functions and dynamics that make the learning possible and in fact very fruitful [4–7]. This challenge has raised the attention of the theoretician community in multiple areas of science. However, despite notable effort to analytically study deep learning [8–15], there are some fundamental questions that are yet to be addressed, for example (i) can we predict practically relevant scores like train and generalization error of deep networks in realistic regimes? (ii) how does information contained in real-world datasets is exploited by the network to extract useful representations (features)?

A great part of the classic theoretical results that we have in machine learning make use of some simple assumption on the training data distribution [16–20]. The physical reason to make such assumption has its roots in the classical statistical mechanics description of disordered systems, mainly spin-glass theory. A fruitful line of research in machine learning, that inherits from disordered systems, indeed aims to compute a partition function that is a *quenched* or *annealed* average over the training data distribution, which is the source of the disorder, allowing to describe the performances of the typical solution independently of the specific dataset used to train the network [21–23]. This analysis, although providing results that hold in full generality, suffers some limitations, mainly (i) the assumption of simple data distributions, which is essential to analytically compute the averaged partition function, is unrealistic when applied to practical settings in machine learning, for example in

computer vision, where the spatial information contained in the dataset is crucial to achieve almost-optimal generalization performance. (ii) averaging over data is very hard when dealing with deep networks. The architectures that are amenable to this kind of study have at most one layer of trainable parameters (i.e. the perceptron, the random features model, and the committee machine).

In my PhD work, as is suggested by the title, I explored a complementary approach to the one discussed above, which does not make any assumption on the structure/distribution of the training data. In this framework, I will show how to derive explicit formulas for the training and generalization error of trained fully-connected deep networks, shallow convolutional and locally-connected networks, in a regime of learning, called *proportional regime*, that assumes the size of the dataset P to be comparable in magnitude to the width of the hidden layers in the model N_ℓ ($\ell = 1, \dots, L$, L being the (finite) depth of the network). The observables that I will show how to compute in this scenario retain an explicit dependence on the training data, since this is never averaged out. Remarkably, it is indeed this dependence that helps to conjecture how the network can operatively exploit the information contained in the trainset to make informed prediction on unseen data, and how this capability is linked to the topology of the network connections. The present work is organized as follows.

In Chapter 1, I will introduce kernel methods, the state-of the art algorithms for object recognition before deep learning, explaining why they still retain a theoretical interest as limiting dynamics of neural networks in a certain regime (the *infinite-width* limit). A crucial link between kernel methods, wide networks and Gaussian Proccesses will also be explored in this chapter, in view of the forthcoming discussion.

Chapter 2 will be dedicated to a class of results on the so-called *infinite-width limit* of neural networks that leverage the same data-agnostic spirit [24–32]. The infinite-width limit is informally defined as the regime where the size of each hidden layer N_ℓ is much larger than the size of the training set P. Here, one shows that the stochastic process that describes information flow in the deep neural network is a familiar Gaussian process (GP), which is completely determined by a non-linear kernel K_L. A fundamental consequence of this finding

is that learning in the infinite-width limit is equivalent to kernel learning [8, 20, 33, 34] with a static kernel K_L that does not evolve during the training dynamics and is completely fixed once the network's weights are initialized. Notably, given the incredibly general nature of these results, GP limits can be derived for virtually any feedforward architecture [29, 35, 36].

In Chapter 3, I will discuss the critical topic of *feature learning* [37–41], i.e. the capability of deep networks to automatically detect useful representations from raw data. This is a fundamental aspect that any minimal theory of deep learning should be able to quantitatively address, and constitutes a limitation of the infinite-width regime, where it is essentially absent [42]. On the contrary, I will show evidence that feature learning occurs and is in fact essential in finite-width convolutional networks, but is almost absent in finite-width standard scaled 1HL fully-connected networks in the proportional regime.

In Chapter 4, I will show how this data-agnostic approach can be extended, using the tools of physics, beyond the infinite-width limit, in particular in the proportional regime introduced above. I will show how a statistical mechanics description is possible in this scenario both for FC networks of arbitrary finite depth, and for shallow networks with local connections, with and without weight sharing.

In Chapter 5, I will try and rationalize the observation made in Chapter 3, through the lense of the framework introduced in Chapter 4. I will show how, thanks the mechanism of *local kernel renormalization*, one can effectively quantify what it means to be "far" from the kernel regime, providing a possible mathematical description of what it means to learn features in neural networks. Inspection of the effective action for a simple architecture with one convolutional HL in the proportional regime, shows a striking difference with respect to the fully-connected case: whereas the FC kernel is just globally renormalized by a scalar parameter, the CNN kernel undergoes a local renormalization, meaning that many more free parameters are allowed to be fine-tuned during training. This finding can be employed to highlight a simple mechanism for feature learning that can take place in finite-width shallow CNNs, but neither in shallow FC architectures nor in LCNs without weight sharing.

Contents

Part I

A tale of data-agnostic theoretical frameworks for deep learning

CHAPTER 1

From kernels to deep learning and back again

Deep learning models have been state-of-the art architectures for many practically-relevant tasks in the last decade. If the reader is familiar with machine learning literature, they probably already read hundreds of similar sentences. For those who are just now approaching the subject, it is worth putting some perspective to this phenomenon, that has already earned the connotation of a revolution. A crucial time marker can be found in 2012, when, for the first time ever, a deep convolutional neural network, AlexNet [1], won the Imagenet competition on object classification [1, 43, 44] (see Figure 1.1). It is fair to say that, after this event, the hype for deep neural networks (DNNs) never went down and still has not to this day.

Before deep learning, the algorithms that reached the best performances on the same kind of tasks were the so-called kernel machines. These are a class of algorithms that use linear classifiers to solve nonlinear problems, by means of a suitable mapping of the train data in a higher dimension. The first and most notable example of these algorithms is the *support vector machine* (SVM), developed by Vapnik and Cortez in 1995 [33]. From the theoretical point of view, a study of the statistical mechanics properties of SVMs was obtained by Dietrich and

collaborators [20] a few years later, in a data-averaged setting.

Even though kernel methods are not used anymore in practice, they recently come back to the attention of the theoreticians, after the discovery that deep neural networks in a certain regime are equivalent to kernel learning with a fixed kernel that depends on the network's weights at initialization and on the train data. This regime, the so-called *infinite-width limit*, will be investigated in great detail in the next Chapter. To better understand the forthcoming discussion, in the rest of this Chapter I will (i) discuss kernel methods and the algorithms that stemmed by them, (ii) introduce deep learning with fully-connected, locally connected and convolutional layers, (iii) analyze the link between kernel methods and and Gaussian processes.

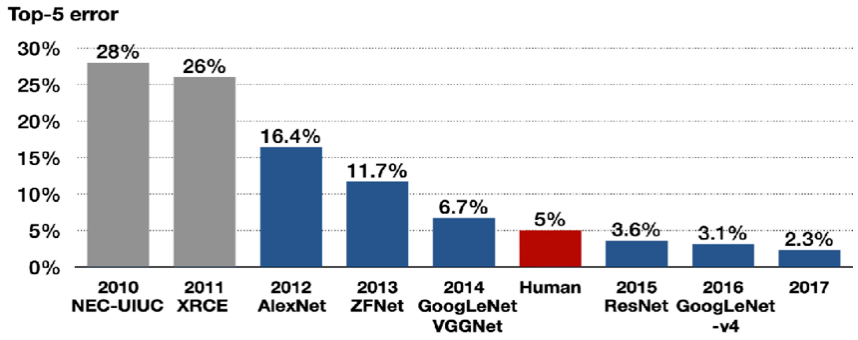


Figure 1.1 –Generalization performance of various architectures in computer vision tasks. Performance reached by winner of the Imagenet competition from 2010 to 2017. The gray bars correspond to kernel based architectures, while blue bars represent the performances of deep convolutional models. The red bar corresponds to human performance, that was surpassed almost ten years ago. The top-5 error refers to the probability that all top-5 classifications proposed by the algorithm for the image are wrong. This image was taken from [45].

1.1 Kernel trick: how to use linear classifiers for non-linear problems

Suppose you have a supervised learning problem with a training set of P elements $\mathcal{D} = \{\mathbf{x}^\mu, y^\mu\}_{\mu=1}^P$, where each vector $\mathbf{x}^\mu \in \mathbb{R}^{N_0}$ has corresponding binary label $y^\mu = \pm 1$. In general,

the separating surface that correctly labels all the datapoints can have any shape, that not necessarily allows analytical investigation. The kernel trick aims at using linear classifiers to attack non-linear problems. The two-step idea is:

1. Embed your data to a higher dimension with a feature map ϕ:

$$\mathbf{x} \in \mathbb{R}^{N_0} \longrightarrow \phi(\mathbf{x}) \in \mathbb{R}^D \qquad D \gg N_0 \tag{1.1}$$

2. Find the optimal separating hyperplane in the embedded space, that is defined by [33] as the linear decision function with maximal margin between the vectors of the two classes (see Figure 1.2).

1.1.1 Max-margin hyperplane

If the dataset $\mathcal{D}$ is linearly separable, then there exist a vector $\mathbf{v}^* \in \mathbb{R}^D$ and a scalar b^* such that, for all the elements in the dataset, the following inequalities hold:

$$\begin{aligned} &\mathbf{v}^* \cdot \phi(\mathbf{x}^\mu) + b^* \geq 1 \qquad \text{if } y^\mu = 1 \\ &\mathbf{v}^* \cdot \phi(\mathbf{x}^\mu) + b^* \leq -1 \quad \text{if } y^\mu = -1, \end{aligned} \tag{1.2}$$

These constraints can be rewritten as a single one:

$$y^\mu(\mathbf{v}^* \cdot \phi(\mathbf{x}) + b) \geq 1 \quad \forall \mu = 1 \dots P \tag{1.3}$$

Therefore, the unique hyperplane satisfying:

$$\mathbf{v}^* \cdot \phi(\mathbf{x}^\mu) + b^* = 0 \tag{1.4}$$

separates the dataset with maximal margin, keeping the largest distance from datapoints x^μ that satisfy Eq. (1.3) with equality. These special datapoints are called *support vectors* of the learning problem. Note that this is equivalent to separating the dataset with a 1HL network

defined as:

$$f^*(\mathbf{x}) = \text{sign}\left(\mathbf{v}^* \cdot \phi(\mathbf{x}) + b^*\right) \tag{1.5}$$

The geometrical distance between the hyperplanes in Eq.(1.2), defined as the margin in [33], is the quantity to be maximized, and is equal to $2/\|\mathbf{v}\|$. Putting everything together, the optimal hyperplane is the solution $\mathbf{v}^* \in \mathbb{R}^D$ to the following minimization problem with P constraints:

$$\begin{aligned} \text{min} \quad & \frac{1}{2}\|\mathbf{v}\|^2 \\ \text{subject to} \quad & y^\mu\, \mathbf{v} \cdot \phi(\mathbf{x}^\mu) \geq 1 \qquad \forall \mu = 1 \dots P \end{aligned} \tag{1.6}$$

Note that the link between margin and robustness (wrt perturbations) of a solution is extensively studied in the literature [12, 46–48].

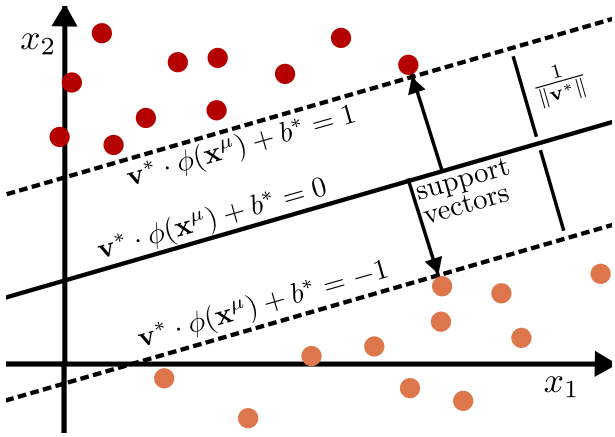


Figure 1.2 –Representation of a 2d linearly separable classification problem. Train examples are 2d vectors depicted with different colors depending on their label (red corresponds to $y = 1$ labeled point, while orange points are labeled $y = -1$). The optimal separating hyperplane (solid line), separates the data with maximal distance between examples of the two classes. This distance, called margin, is given by the inverse norm of optimal solution v^* (see main text). The examples that determine the margin (ie, the closest to max-margin hyperplane) are called support vectors.

1.1.2 Making predictions with support vector machines

The previous minimization problem, referred to as *primal* problem, requires finding a vector $\mathbf{v}^* \in \mathbb{R}^D$. Since the embedding dimension $D \gg N_0$ is typically large, it is convenient to write the problem in the *dual* space, where the minimization requires finding P scalar parameters. To do so, one should write the Lagrange function introducing P multipliers (one for each constraint in standard form):

$$\mathcal{L} = \frac{1}{2}\|\mathbf{v}\|^2 + \sum_\mu \alpha_\mu \left[1 - y_\mu \ (\mathbf{v} \cdot \phi(\mathbf{x}^\mu) + b)\right] \qquad \text{with} \ \ \alpha_\mu \geq 0 \ \ \forall \mu, \tag{1.7}$$

Solving $\partial\mathcal{L}/\partial v_i = 0$ yields optimal $\mathbf{v}^*$ as a function of the dual variables α_μ:

$$\mathbf{v}^* = \sum_\mu \alpha_\mu y_\mu \phi(\mathbf{x}^\mu) \tag{1.8}$$

In the dual space, the problem becomes the maximization of the Lagrangian

$$\mathcal{L} = \sum_\mu \alpha_\mu \left(1 - b y^\mu\right) - \frac{1}{2}\sum_{\mu\nu} \alpha_\mu \alpha_\nu \, y_\mu y_\nu K(\mathbf{x}^\mu, \mathbf{x}^\nu), \tag{1.9}$$

with respect to the multipliers α_μ. Note that the dual problem is computationally convenient with respect to the primal one because, in general, the size of the trainset P is much smaller than the embedding dimension D. A support vector machine solves this quadratic programming optimization to find the multipliers α_μ. The real-valued function $K(\mathbf{x}, \mathbf{x}')$ is called hereby a *kernel function*:

$$K(\mathbf{x}, \mathbf{x}') \equiv \phi(\mathbf{x}) \cdot \phi(\mathbf{x}') \tag{1.10}$$

The $P \times P$ matrix with elements $K^{\mu\nu} = \phi(\mathbf{x}^\mu) \cdot \phi(\mathbf{x}^\nu)$ will be denoted *kernel matrix* in the following. Note that K depends explicitly on the dataset, and on the choice of the mapping ϕ.

From Eq. (1.8), it is clear that the optimal hyperplane is a combination of the train examples in the feature space with some coefficients given by the dual variables α_μ. Moreover, the

Karush–Kuhn–Tucker (KKT) conditions impose some constraints on the α_μ:

$$\alpha_\mu \left(y_\mu \, \mathbf{v}^* \cdot \phi(\mathbf{x}^\mu) - 1 \right) = 0 \qquad \forall \mu = 1 \dots P, \tag{1.11}$$

implying that, whenever the primal variable, or constraint $y^\mu \, \mathbf{v}^* \cdot \phi(\mathbf{x}^\mu) \geq 1$, is satisfied with equality, the correspondent dual variable α_μ will be nonzero and vice versa. We can write:

$$\mathbf{v}^* = \sum_{\mu \in \text{support vectors}} \alpha_\mu y_\mu \, \phi(\mathbf{x}^\mu) \tag{1.12}$$

where the support vectors are the dataset elements $\mathbf{x}^\mu$ for which $\alpha_\mu > 0$ by the KKT conditions (1.11). If everything but the support vectors were discarded by the trainset, the network with weights (1.12) would still correctly classify the rest of the dataset. The decision boundary of the SVM is given in terms of the kernel matrix:

$$f^*(\mathbf{x}) = \text{sign} \left(\sum_{\mu \in \text{support vectors}} \alpha_\mu K(\mathbf{x}, \mathbf{x}^\mu) + b \right) \tag{1.13}$$

where K is defined in Eq. (1.10). Note that the kernel completely determines the SVM output in a data-agnostic way, in the sense that no assumption is needed on the data distribution, and the information contained in the data is explicitely found in the network's function (and therefore in all the other relevant observables).

1.2 Deep Learning: brief history and definitions

The journey of artificial networks begins in the late 1950s, when Frank Rosenblatt introduced the world to the first feedforward neural network: the perceptron, a single-layer neural network designed to mimic a neuron's binary response [49]. Since then, perceptrons have been stuck together in many different ways, and have been of inspiration to build a plenitude of models with different geometries, what we now call deep neural networks. On the algorithmic side, crucial advancement was brought by the development of the backpropagation algorithm

[50], and its implementation in the 1980s [51], that allowed the training of deeper and more complex models.

Starting from the 2010s, the boost of available computational power and the proliferation of large datasets allowed the rise of specialized neural network architectures tailored for specific data types. This breakthrough was exemplified in 2012, with AlexNet hands down won the Imagenet competition [43]. By the late 2010s, a groundbreaking architecture known as the transformer, equipped with attention mechanisms, reshaped the landscape of natural language processing. Introduced by Vaswani and his team in 2017 [3], this design was adept at capturing relationships in data irrespective of the distance between data points. This laid the foundation for monumental models such as OpenAI's GPT and Google's BERT, that already become common instruments to ease our everyday tasks.

All these incredible technological advancement that we are witnessing thanks to deep learning still lacks a satisfactory theoretical foundation. As sometimes happens in history, theoreticians have not been able to keep up with the intuition of practitioners.

The overarching goal of the present work is indeed that to develop a general method to analytically study feedforward models, in a regime of learning (tradeoff between number of parameters and dataset size) that is less idealised than what we currently have. To this end, let us now dive more deeply in the mathematical representation of neural networks that will be studied in the following.

1.2.1 Fully-connected architectures

Fully-connected (FC) deep networks (multilayer perceptrons) are the simplest deep feedforward architectures [1].

We define deep neural networks $f_{\text{FCN}}(\mathbf{x})$ with L FC hidden layers (HLs) recursively, with the pre-activations of each layer $h_{i_\ell}^{(\ell)}$ ($i_\ell = 1, \dots, N_\ell$; $\ell = 1, \dots, L$) are given recursively as a

[1] In a feedforward network, the signal propagates only in one direction, meaning that information never goes back to a previous layer.

non-linear function of the pre-activations at the previous layer $h_{i_{\ell-1}}^{(\ell-1)}$ ($i_{\ell-1} = 1, \dots, N_{\ell-1}$):

$$
\begin{aligned}
h_{i_\ell}^{(\ell)} &= \sum_{i_{\ell-1}=1}^{N_{\ell-1}} W_{i_\ell i_{\ell-1}}^{(\ell)} \sigma\left(h_{i_{\ell-1}}^{(\ell-1)}\right) + b_{i_\ell}^{(\ell)} , \\
h_{i_1}^{(1)} &= \sum_{i_0=1}^{N_0} W_{i_1 i_0}^{(1)} x_{i_0} + b_{i_1}^{(1)}
\end{aligned}
\tag{1.14}
$$

where $W^{(\ell)}$ and $b^{(\ell)}$ are respectively the weights and the biases of the ℓ-th layer, whereas the input layer has dimension N_0 (the input data dimension). σ is a non-linear activation function and it is common to each layer. We add one last readout layer and we define the scalar function implemented by the deep convolutional network as:

$$
f_{\text{FCN}}(\mathbf{x}) = \sum_{i_L=1}^{N_L} v_{i_L} \sigma\left[h_{i_L}^{(L)}(\mathbf{x})\right] + b_{i_L} , \tag{1.15}
$$

where $\mathbf{v} = (v_1, \dots, v_{i_L})$ is the vector of weights of the last layer.

1.2.2 Convolutional neural networks

Inspired by the mammal visual cortex [52, 53], CNNs represent a clever computational way to implement traslational invariance, which is critical to extract information from visual patterns. The two fundamental ingredients of a convolutional layer are *local connectivity* and *weight sharing*: whereas a given neuron in a FC hidden layer receives input from all the neurons in the previous layer, neurons in a CNN are arranged in a d-dimensional array that reflects the corresponding d-dimensional arrangement of the input data (e.g. $d = 2$ for images). Each neuron in a given layer here interacts only with a local neighbourhood of neurons in the previous layer, in a translational invariant way implemented via a shared (usually small) d-dimensional mask of learnable weights. These operations define a single convolutional channel that takes as input a d-dimensional array and outputs another d-dimensional array, whose dimensions are determined by technical details (such as the stride, padding and dimension of the filter mask). Usually many of these channels are piled up in a convolutional layer to form, overall,

a $(d+1)$-dimensional array.

Let me now define an architecture with L hidden convolutional layers. For simplicity I will restrict the definition and analysis to one-dimensional convolutions, but the model can be easily generalized to d-dimensional convolutions, paying a price in terms of heaviness of the notation. Pre-activations in the hidden layers are given by:

$$h_{i,a}^{(\ell)}(\mathbf{x}) = \sum_{b=1}^{C_\ell - 1} \sum_{m=-\lfloor M_\ell/2 \rfloor}^{\lfloor M_\ell/2 \rfloor} W_{m,b}^{(\ell)} \phi_{S_\ell i+m,a+b}^{(\ell)} + b_a^{(\ell)} \,. \tag{1.16}$$

$$\phi_{i,a}^{(\ell)} = \begin{cases} x_{i,a} & \ell = 1 \\ \sigma\left(h_{i,a}^{(\ell-1)}\right) & \ell > 1 \end{cases}$$

Here σ is a odd activation function, M_ℓ is the dimension of the channel mask at layer ℓ, S_ℓ is the stride, the index $i = 1, \ldots, \lfloor N_0/S \rfloor$ runs over the input coordinates, the index $b = 1, \ldots, C_\ell$ runs over the channels and the index m moves through the spatial mask of the convolutional filter. For simplicity we define periodic boundary conditions (PBCs) over the input coordinates and we consider odd values of M. As for the 1HL FC network we add one readout layer, therefore the function implemented by the CNN is given by:

$$f_{\text{CNN}}(\mathbf{x}) = \sum_{i=1}^{N_L} \sum_{a=1}^{C_{L-1}} v_i^a \sigma\left[h_{i,a}^{(L)}(\mathbf{x})\right] \,, \tag{1.17}$$

where the v's are the learnable weights of the readout layer, indicised by $i = 1 \ldots N_L$, with N_L being the number of neuron in the readout layer. In this setting with PBCs, N_L satisfies the following recursion $N_\ell = \lfloor N_{\ell-1}/S \rfloor$.

1.2.3 Scaling and initialization schemes for neural network training

The supervised training process of a neural network on a regression task is generally carried out with two ingredients:

- A collection of examples, the trainset $\mathcal{D} = \{\mathbf{x}^\mu, y^\mu\}_{\mu=1}^P$, where each N_0-dimensional

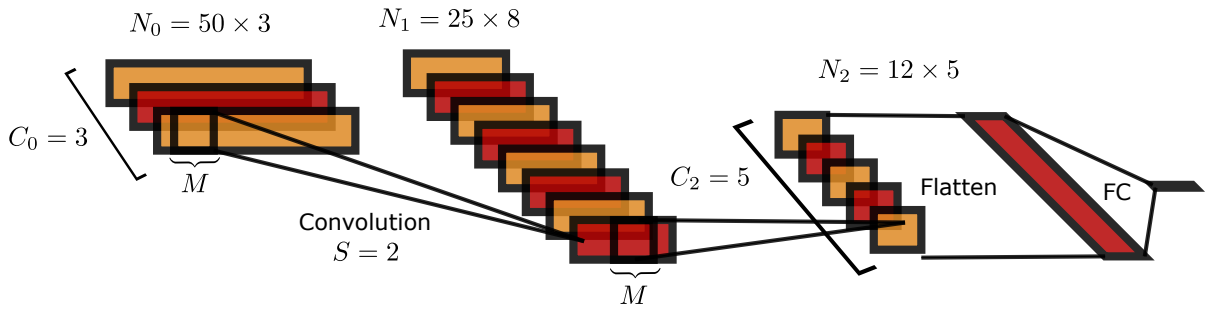


Figure 1.3 –One-dimensional CNN. A visual representation of a 1d CNN with two hidden layers ($L = 2$) described by Equations (1.17) (1.16). From left to right: the filter mask of size $M = 10$ parses the input with stride $S = 2$. Periodic boundary conditions ensure that $N_\ell = \lfloor N_{\ell-1}/2 \rfloor$. After the second convolution, the activations are arranged in a single vector (flatten operation) and a final readout FC layer produces a one-dimensional output y.

datapoint $\boldsymbol{x}^\mu \in \mathbb{R}^{N_0}$ has a corresponding scalar label $y^\mu \in \mathbb{R}$. In the following I will use the notation $\mathbf{y} = (y^\mu \in \mathcal{D})$, and $X = (\mathbf{x}^\mu \in \mathcal{D})$, respectively for the ensemble of labels and datapoints.

- A cost -or loss- $\mathcal{L}(\theta)$, which is a function of all the network's parameters, collectively indicated as $\theta = \{W^{(\ell)}\}_{\ell=1}^{L}$, that one wishes to minimise wrt the trainset.

One popular loss function, with a meaningful interpretation in terms of Bayesian learning that makes it crucial in this work, is the mean squared error (MSE):

$$\mathcal{L}_{\text{MSE}}(\theta) = \frac{1}{2}\sum_{\mu=1}^{P}\left(y^\mu - f_\theta(\mathbf{x}^\mu)\right)^2 \tag{1.18}$$

Although this loss is designed for regression problems, note that it can equivalently address classification problems, with a suitable mapping of the class label into a scalar.

Usually, the Loss minimization is operatively achieved with gradient-based methods, meaning that the parameters are updated in the inverse direction of the Loss gradient wrt the weigths:

$$\theta(t+1) = \theta(t) - \eta \nabla_\theta \mathcal{L}(\theta)|_{\theta=\theta(t)} \tag{1.19}$$

where η is a learning rate. This update rule describes the vanilla full-gradient descent (GD),

that is known to struggle when the energy landscape is rugged [48, 54, 55]. Many variants of this algorithm exist (eg. stochastic gradient descent SGD, ADAM, entropy-SGD) that aim at overcoming the limitations of the vanilla GD, making gradient-based methods state-of-the-art algorithms to train neural networks [56–58]. When such a learning scheme is chosen, it is fundamental to also tune the scaling of all the quantities that might affect training dynamics. This topic is discussed both among theoreticians and practitioners, and there exists different initialization schemes that are known to help dynamics to converge faster [59] or to better generalizing solutions [60]. A theoretical categorization of network's parameterizations was recently carried out in [61]. The authors define a natural space for network parameterization that involves scaling of the following three quantities:

1. The function $f(\mathbf{x})$, through the scaling of its parameters, for FC layers we have

$$W_{ij}^{(\ell)} \to N_\ell^{-a_\ell} W_{ij}^{(\ell)} \tag{1.20}$$

   while for convolutional layers one should replace N_ℓ with the mask size $M^{(\ell)}$:

2. The initialization of the parameters, or prior over the parameters, that is taken Gaussian distributed:

$$W_{ij}^{\ell} \sim \mathcal{N}\left(0, \frac{N_\ell^{-b_\ell}}{\lambda_\ell}\right), \tag{1.21}$$

   where $\mathcal{N}(\mu, \sigma^2)$ denotes a normalized Gaussian with mean μ and variance σ. The factor λ_ℓ, called Gaussian prior, has dimension of an inverse variance.

3. The learning rate η:

$$\eta \to N_\ell^{-c} \eta \tag{1.22}$$

The set of variables $\{a_\ell, b_\ell, c_\ell\}$ completely defines the chosen parameterization, for this reason

Parametrization	***a*** parameters	***b*** prior	***c*** learning rate
NTK (NTP)	$a_\ell = 1/2$	$b_\ell = 0$	$c = 0$
Mean Field (MFP)	$a_\ell = 1$	$b_l = 0$	$c = -1$
Standard (SP)	$a_\ell = 0$	$b_l = \frac{1}{2}$	$c = 0$

Table 1.1 –Comparison of popular parametrizations for FC neural networks. The parameters a are associated with the scaling of the network function, through its weights $N_\ell^{-a_\ell} W_{ij}^{(\ell)}$. The parameters b refer to the initialization of parameters $W_{ij}^{\ell} \sim \mathcal{N}\left(0, N_\ell^{-b_\ell}/\lambda_\ell\right)$. The third column parameters c control the scaling of the learning rate $N_\ell^{-c}\eta$.

the authors indeed call this schemes abc-parameterizations. The networks I will investigate in this work will be initialised with the so-called standard NTK parameterization, firstly presented by Neal in the 90's [24], although with a different name. For a FC architecture, in terms of abc parameterization, the NTK is obtained taking:

$$a_\ell = 1/2 \qquad b_\ell = 0 \ \forall \ell \tag{1.23}$$

It is very important to stress that different scalings lead to different properties and behaviours both from the theoretical and algorithmic properties of training and solutions found. A summary of the most used initialization schemes can be found in Table 1.1.

1.2.4 Small remark: goal of training a network

Suppose you successfully minimized your loss function, and now you have a configuration θ^* of optimal parameters. What does one want to do with θ^*? In general, one whishes to know if the network managed to *extrapolate* some information from the training process, and therefore is able to correctly classify (or predict the scalar label) of an *unseen* example $\mathbf{x}^*$, with correct label y^*, that is not contained in the trainset $\mathbf{x}^* \notin \mathcal{D}$. In some contests, I will alternatively use the notation $(\mathbf{x}^0, y^0)$ for the unseen example.

1.2.5 Bayesian training of neural networks

Suppose you want to use your trainset $\mathcal{D}$ to build a Bayesian inference model to make predictions for unseen examples $(\mathbf{x}^*, y^*)$. The model that we are interested in here is of course a neural network, that implements the generic function $f_\theta(\mathbf{x})$, where θ denotes the ensemble of network's parameters with prior distribution $p(\theta)$. The optimal Bayesian estimator for the parameters θ^* is the one that maximizes the following conditional log-likelihood:

$$\theta^* = \underset{\theta}{\text{argmax}}\ \log p(\mathbf{y}|X,\theta)p(\theta). \tag{1.24}$$

That is, the model that is more likely to yield the vector $\mathbf{y}$ of outputs in response to the P train examples X. If the examples are iid, the previous probability distribution factorizes:

$$\theta^* = \underset{\theta}{\text{argmax}} \sum_{\mu=1}^{P} \log p(y^\mu|\mathbf{x},\theta) + \log p(\theta) \tag{1.25}$$

With a deterministic network that is expressive enough to perfectly interpolate the dataset, the distribution $p(y|\mathbf{x})$ is a delta function:

$$p(y|\mathbf{x}) = \delta(y - f(\boldsymbol{x})) \tag{1.26}$$

More in general, one could consider a noisy network (with Gaussian-distributed noise), resulting in the following distribution:

$$p(y|\mathbf{x}) = \mathcal{N}_y(f(\mathbf{x}),\ 1/\beta) \tag{1.27}$$

where the notation $\mathcal{N}_y(\mu, \Sigma)$, recurrent in the present work, always indicates a well normalized Gaussian on the variable y, with mean μ and variance (covariance matrix) Σ. Note that, as is should be, the noisy case reconduces to the noisless one in the $\beta \to \infty$ limit. If the parameters are taken iid Gaussian $\mathcal{N}(0, 1/\lambda)$ at initialization, the optimal parameter can be

found maximizing the following expression:

$$\theta^* = \underset{\theta}{\text{argmax}} \left[\frac{P}{2} \log \beta - \frac{P}{2} \log(2\pi) - \sum_{\mu=1}^{P} \frac{\beta}{2} \left(y^\mu - f_\theta(\mathbf{x}^\mu) \right)^2 - \lambda_1 \|\theta\|^2 \right] \tag{1.28}$$

which yields the same result as the minimization of the MSE in Eq. (1.18) in the $\beta \to \infty$ limit. Even at finite β, this maximization problem is found to be equivalent to the statistical mechanics description that I am going to explore, giving it a nice Bayesian interpretation.

As a standard practice in statisical mechanics, let me define the partition function associated to the MSE in (1.18):

$$Z = \int D\theta \, e^{-\beta \mathcal{L}(\theta)} \, . \tag{1.29}$$

where the symbol $\int D\theta$ indicates the collective integration over the weights of the network, with their prior Gaussian distribution. This choice enforces minimization of the training MSE for $\beta \to \infty$, preserving the effect of the Gaussian prior in the Bayesian setting.

All the observables $\langle O \rangle$ are computed as averages over the Gibbs ensemble:

$$\langle O \rangle = \int D\theta \, O(\theta) \frac{e^{-\beta \mathcal{L}(\theta)}}{Z} \tag{1.30}$$

are averages over samples from the Gibbs posterior distribution at temperature $1/\beta$. The average training error at a given inverse temperature β is given by:

$$\langle \epsilon_{\text{t}} \rangle = \frac{1}{P} \int D\theta \, [\mathcal{L}(\theta) - \mathcal{L}_{\text{reg}}(\theta)] \frac{e^{-\beta \mathcal{L}(\theta)}}{Z} \, , \tag{1.31}$$

while the average test error over a new (unseen) example $(\mathbf{x}^*, y^*)$ reads:

$$\langle \epsilon_{\text{g}}(\mathbf{x}^*, y^*) \rangle = \int D\theta \, [y^* - f_\theta(\mathbf{x}^*)]^2 \frac{e^{-\beta \mathcal{L}(\theta)}}{Z} \, . \tag{1.32}$$

Operatively, samples from the Gibbs posterior are obtained training the network with a Langevin dynamics, that corresponds to the following update rule for the network's parame-

ters:

$$\theta(t+1) = \theta(t) - \eta \nabla_\theta \mathcal{L}(\theta(t)) + \sqrt{2T\eta}\epsilon(t) \tag{1.33}$$

where $T = 1/\beta$ is the temperature, η is the learning rate, $\epsilon(t)$ is a white Gaussian noise vector with entries drawn from a standard normal distribution. Note that this training is quantitatively different from the commonly used gradient descent (GD) and stochastic gradient descent (SGD) algorithms. The former is a deterministic algorithm, concretely obtained by removing the noise dependent term from Eq. (1.33). The latter is a noisy version of GD, where, however, one does not have control over the non-Gaussian noise. Both of these algorithms cannot be used if one wishes to systematically sample from an equilibrium distribution of the weights.

1.3 Gaussian processes and their link to standard-scaled wide networks: how kernels survived the deep learning revolution

As anticipated in the introduction, kernel methods are outdated in practice, but they still retain some interest from the theoretical point of view. In particular, thanks to the crucial discovery that wide deep networks are equivalent to Gaussian Processes (GP). The pioneering paper in which this equivalence is highlighted for the first time in 1HL networks was written by Neal in 1994 [24], whose result I will firstly report for historical and pedagogical reasons. Neal describes a 1HL architecture with a bounded activation function $\sigma = \tanh$, defined as:

$$f_{1\text{HL}}(\mathbf{x}) = \sum_{i_1=1}^{N_1} v_{i_1} \tanh\left(h_i(\mathbf{x})\right), \qquad h_i(\mathbf{x}) = \sum_{i_0=1}^{N_0} W_{i_0 i_1} x_{i_0}, \tag{1.34}$$

where the weights are initialized as $v_i \sim \mathcal{N}(0, \lambda_1/N_1)$ and $W_{ij} \sim \mathcal{N}(0, \lambda_0/N_0)$, that is with the standard initialization discussed in Section 1.2.3, which is a crucial choice to obtain the GP

equivalence. Let me put the gaussian priors $\lambda_1 = \lambda_0 = 1$ for the moment, I will re-integrate them in the end. It follows intuitively from the central limit theorem (CLT) that, with the standard parametrization in Table 1.1, the experimental mean and covariance of the output distribution become deterministic in the limit $N_1 \to \infty$ and P fixed [2], with:

$$\bar{f}_{1\text{HL}}(\mathbf{x}) \underset{N_1\to\infty}{\to} \langle f(\mathbf{x})\rangle_{P(h_i)} = 0 \tag{1.35}$$

$$\text{cov}\big(f(\mathbf{x}),\, f(\mathbf{x}')\big) \underset{N_1\to\infty}{\to} \langle \sigma\left(h_i(\mathbf{x})\right) \sigma\left(h_i(\mathbf{x}')\right)\rangle_{P(h_i)}, \tag{1.36}$$

where the expected value of a function $f(\mathbf{x})$ is deifned in a standard way as:

$$\langle f(\mathbf{x})\rangle_{p(\{x_n\})} = \int D\mathbf{x}\, f(\mathbf{x}), \tag{1.37}$$

having used a short notation $D\mathbf{x} = p(\{x_n\}) \prod_n \mathrm{d}x_n$ for the integration over all the coordinates x_n of vector $\mathbf{x}$, and their joint probability distribution. I can and will instantly drop the index i, since the $\{h_i\}$ are independent. Note that the distribution of the pre-activations $p(h)$ depends solely on the inputs $\mathbf{x}$, and only through the two-point correlation function:

$$\bar{h}(\mathbf{x}) \underset{N_1\to\infty}{\to} \bar{\mathbf{x}} \tag{1.38}$$

$$\text{cov}\left(h(\mathbf{x}),\, h(\mathbf{x}')\right) \underset{N_1\to\infty}{\to} \frac{1}{\lambda_0}\text{cov}(\mathbf{x},\, \mathbf{x}') \tag{1.39}$$

Choosing a collection of examples $X = (\mathbf{x}^\mu,\, \mu = 1, \dots, P)$, where each $\mathbf{x}^\mu \in \mathbb{R}^{N_0}$, one can analyze the joint distribution of the outputs $\mathbf{f} = (f(\mathbf{x}^\mu),\, \mathbf{x}^\mu \in X)$ and pre-activations $\mathbf{h} = (h(\mathbf{x}^\mu),\, \mathbf{x}^\mu \in X)$. The probability distribution $P(\{\mathbf{h}\})$ can be expressed as a function of the two-point correlation matrix of the dataset C:

$$p(\{\mathbf{h}\}) = \mathcal{N}_{\mathbf{h}}\left(0, C\right) \equiv \frac{e^{\sum_{\mu\nu} h^\mu C^{-1}_{\mu\nu} h^\nu}}{\sqrt{2\pi \det C}}, \qquad C_{\mu\nu} = \frac{\mathbf{x}_\mu \cdot \mathbf{x}_\nu}{N_0}. \tag{1.40}$$

[2] First appearance of what will be known as *infinite-width limit*

The empirical covariance between two outputs, in the large N_1 limit reads, remembering Eq. (1.36):

$$\mathrm{cov}\big(f(\mathbf{x}^\mu)f(\mathbf{x}^\nu)\big) \underset{\text{def}}{=} \tilde{K}(\mathbf{x}^\mu,\mathbf{x}^\nu) \underset{N_1\to\infty}{\to} \langle \sigma(h(\mathbf{x}^\mu))\sigma(h(\mathbf{x}^\nu))\rangle_{P(\{\mathbf{h}\})} \underset{\text{def}}{=} K(\mathbf{x}^\mu,\mathbf{x}^\nu), \quad (1.41)$$

where $\tilde{K}$ is the empirical Kernel, while K is the so-called *neural network Gaussian process* (NNGP) deterministic kernel, extensively found in the literature [24, 27]. Taking the Gaussian priors into account amounts to substituting $C \to C/\lambda_0$, and $K \to K/\lambda_1$,

In the same paper, Neal also conjectures that similar considerations on the output's prior distribution should hold for deeper networks, still in the case where the dataset size P is fixed, the hidden layers have an infinite number of neurons, and the standard initialization is employed. Putting this intuition on theoretical grounds took two decades, but finally in 2018 a theorem for prior (and posterior) distributions of wide standard-scaled networks was finally written [27], as we will see in the next Chapter.

1.3.1 Remarks on the NNGP kernel and notation

It can be shown that each of the integrals in Eq. (1.41) can be reduced to a two-dimensional integral (see Appendix B):

$$K(\mathbf{x},\mathbf{x}') = \int \mathrm{d}\mathbf{t}\, N_{\mathbf{t}}(0,\Sigma)\,\sigma(t_1)\sigma(t_2), \qquad \Sigma = \frac{1}{\lambda_0 N_0}\begin{pmatrix} \mathbf{x}\cdot\mathbf{x} & \mathbf{x}\cdot\mathbf{x}' \\ \mathbf{x}\cdot\mathbf{x}' & \mathbf{x}'\cdot\mathbf{x}' \end{pmatrix}, \quad (1.42)$$

where I have used the vector notation $\mathbf{t} = (t_1, t_2)$.

The kernel function K, that is in principle a two-point function, can be seen as an operator that transforms positive semi-definite (PSD) matrices into PSD matrices of the same size. Therefore, sometimes I will make use of the notation $K(C)$ for a $P \times P$ matrix, that is:

$$[K(C)]_{\mu\nu} \equiv K_{\mu\nu} = \int \mathrm{d}t_1\ \mathrm{d}t_2\, N_{\mathbf{t}}(0,\tilde{C})\,\sigma(t_1)\sigma(t_2), \qquad \tilde{C} = \begin{pmatrix} C_{\mu\mu} & C_{\mu\nu} \\ C_{\mu\nu} & C_{\nu\nu} \end{pmatrix}, \quad (1.43)$$

the form of the NNGP kernel only depends on the activation function σ. For various commonly-used activation functions, the previous integral can be computed analytically. In Appendix F.1 I report the exact expressions of the Kernel functions for $\sigma = \text{ReLU}$ and $\sigma = \text{Erf}$.

1.3.2 Bayesian posterior over outputs from Gaussian process

More formally, we have found an infinite collection of variables -i.e. all the possible infinite-width network outputs - with the property that every finite subset $\mathbf{f} = (f(\mathbf{x}^\mu),\, \mathbf{x}^\mu \in X)$ is jointly multivariate-Gaussian distributed:

$$p\left(\mathbf{f}\right) = \mathcal{N}(0, K_{X,X}), \tag{1.44}$$

where the matrix $K_{X,X}$ comprises all the covariances between the elements in X:

$$K_{X,X} = \left(K(\mathbf{x}^\mu, \mathbf{x}^\nu),\, \mathbf{x}^\mu, \mathbf{x}^\nu \in X\right). \tag{1.45}$$

This is precisely the definition of a Gaussian Process (GP) $\mathcal{GP}(0, K)$.

Definition 1. *A Gaussian process $\mathcal{GP}(\mu, \Sigma)$ is a (potentially infinte) collection of variables $\{k_n\}$ such that the joint distribution of every finite subset of these variables is multivariate Gaussian with mean μ and covariance Σ:*

$$\{k_n\} \sim \mathcal{GP}(\mu, \Sigma) \tag{1.46}$$

This consideration allows to quickly find the predictive *posterior* distribution for the output of the network to the new set of examples X^*, such that $X \cap X^* = \emptyset$, that is $\mathbf{f}^*$ (see for example [62]). Let me denote $\mathbf{y}$ the vector of network's targets to the dataset inputs X. $\mathbf{y}$ are the *observations* (in the GP terminology) relative to the data in X, that I have to condition the prior on, to get the posterior. Since we are dealing with a GP, we can easily write the joint

probability distribution $P(\mathbf{f}^*, \mathbf{y}|X, X^*)$ that is Gaussian, and reads:

$$\begin{bmatrix} \mathbf{y} \\ \mathbf{f}^* \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mathbf{0} \\ \mathbf{0} \end{bmatrix}, \begin{bmatrix} K_{X,X} & K_{X,X^*} \\ K_{X,X^*} & K_{X^*,X^*} \end{bmatrix} \right), \tag{1.47}$$

where

$$K_{X,X^*} = (K(\mathbf{x}^\mu, \mathbf{x}^*),\ \mathbf{x}^\mu \in X, \mathbf{x}^* \in X^*) \tag{1.48}$$

$$K_{X^*,X^*} = (K(\mathbf{x}^*, \mathbf{x}^*),\ x^* \in X^*)\,. \tag{1.49}$$

with the Kernel function $K(\cdot,\cdot)$ defined in (1.42). The desired predictive conditional probability distribution reads:

$$p\left(f^*|\mathbf{y}, \mathbf{x}^*, X\right) = \int p\left(f^*, \mathbf{f}|\mathbf{x}^*, X\right) p(\mathbf{y}|\mathbf{f})\, \mathrm{d}\mathbf{f} \tag{1.50}$$

From the expression (1.47), it is straightforward to compute the previous integral in both the noisy and noiseless case (see equations (1.26) and (1.27)):

$$P(\mathbf{f}^*|\mathbf{y}) = \mathcal{N}\left(\gamma,\ \Sigma\right) \tag{1.51}$$

$$\gamma = K_{X,X^*}\left[K_{X,X} + \frac{\mathbb{1}}{\beta}\right]^{-1}\mathbf{y} \tag{1.52}$$

$$\Sigma = K_{X^*,X^*} - K_{X,X^*}\left[K_{X,X} + \frac{\mathbb{1}}{\beta}\right]^{-1}K_{X^*,X}. \tag{1.53}$$

Our best predictions for the new examples in X^* is the vector γ, and the uncertainty on this measure will be given by the covariance matrix Σ. Note that this is a non-parametric model for inference, since the output for unseen examples and all the related observables do not depend on the network's weights θ.

Without further ado we can move to the next Chapter to dive into the exploration of infinite-width networks with the standard scaling.

CHAPTER 2

The infinite-width limit of deep neural networks with the standard scaling

The infinite-width limit of neural networks, which has been studied for more than twenty years, is defined as the limit where the size of each hidden layer N_ℓ is taken to be large while the size of the trainset P is kept fixed. The first results from R. Neal on 1HL architectures date back to 1996 [24, 25], where the author conjectures that similar results should hold for deeper networks as well. More recently, a rigorous theory was developed for deep fully-connected, [26–31], convoltuional [29, 32], networks with attention layers [35] and is shown to extend to virtually any feedforward architecture [63]. In this regime, independently of what dynamics is chosen for training, all the generalization capabilities of the network are completely determined by a fixed kernel that does not evolve during training. The specific form of the kernel that arises, however, does depend on the architecture and on the learning dynamics, as I will show in the following. The important caveat that I want again to stress, is that all these results are obtained considering the standard NTK parameterization. Different initalization schemes lead to different results, see for instance [36, 64–68].

For the fully-connected network defined in Eqs. (1.15) and (1.14), the standard NTK parame-

terization corresponds to taking random Gaussian weights and biases as:

$$W_{ij}^{\ell} \sim \mathcal{N}\left(0, \frac{1}{\lambda_\ell \, N_\ell}\right) \qquad b^{\ell} \sim \mathcal{N}\left(0, \frac{1}{\lambda_b}\right) \qquad \forall \ell = 1 \dots L. \tag{2.1}$$

The factor λ_ℓ, called Gaussian prior, has dimension of an inverse variance.

2.1 Large-width limit in the Bayesian setting

In this section I will analyse the infinite-width limit of NTK scaled FC networks in the Bayesian setting described in Section 1.2.5, highliting their equivalence with Gaussian processes. With the choice of Bayesian training, scaling the network's parameters or choosing a particular initialization scheme for the weights is identical, because both operations affect the posterior distribution (that we want to sample) in the same way. In particular, referring to Table 1.1, NTK and standard parameterization are equivalent. For the sake of brevity, I will limit my analisys to deep FCNs and CNN/LCNs.

2.1.1 Prior distribution over outputs in fully-connected networks with finite depth

Here I will present the general statement of equivalence between the prior over functions of a wide FCN with an arbitrary (but finite) number of hidden layers L, and a Gaussian process. This statement was formulated as a theorem in 2018 by [26], twenty years after being conjectured by Neal [24]. In proving these results, I will follow the pedagogic presentation of [69]. The following theorem holds for more general networks than those defined in (1.15). In particular, we can modify the readout layer to allow the network to have multiple outputs, namely:

$$h_i^{(L+1)} \qquad i = 1 \dots N_{L+1} \tag{2.2}$$

where N_{L+1} is the number of the network's outputs. The main result of [27] can be expressed as the following theorem.

Theorem 1 (Equivalence between FC NNs and GPs in the infinite-width limit.). *At fixed length L, with a polynomially bounded activation function $\sigma : \mathbb{R} \to \mathbb{R}$, the collection of random variables $\{h_i^{L+1}\}$ with the standard scaling in Eq.* (2.1) *converges in distribution to a Gaussian process:*

$$\lim_{\{N_\ell\}\to\infty} \{h_i^{(L+1)}\} \to \mathcal{GP}\left(0, K^{(L)}\right)^{\otimes N_{L+1}} \tag{2.3}$$

The GP variance is the L-th layer kernel matrix $K^{(L)}$. We have:

$$\lim_{\{N_\ell\}\to\infty} \operatorname{cov}\left(h_i^{(\ell+1)}(\mathbf{x}^\mu) h_j^{(\ell+1)}(\mathbf{x}^\nu)\right) = \delta_{ij}\left[K^{(\ell)}\right]_{\mu\nu} \tag{2.4}$$

with K^ℓ given by a recurrence relation:

$$K^{(\ell+1)} = \frac{1}{\lambda_b} + \frac{1}{\lambda_\ell}\left\langle \sigma\left(h^{(\ell+1)}(\mathbf{x}^\mu)\right)\sigma\left(h^{(\ell+1)}(\mathbf{x}^\nu)\right)\right\rangle_{\mathcal{N}(0,K^{(\ell)})} \tag{2.5}$$

$$= \frac{1}{\lambda_b} + \frac{1}{\lambda_\ell} K\left(K^{(\ell)}\right), \qquad K^{(0)} \equiv C. \tag{2.6}$$

with $K(\cdot)$ is defined in Eqs. (1.42) *and C is the covariance matrix of the dataset with elements $C_{\mu\nu} = \mathbf{x}^\mu \mathbf{x}^n u/\lambda_0 N_0$*

Proof. Let us consider the collection of train examples $X = \{\mathbf{x}^\mu\}_{\mu=1}^P$. The results of the application of the network in Eq. (1.15) with the scaling in Eq. (2.1) can be collected into a vector $\mathbf{h}_i^{(\ell)} = \left(h_i^{(\ell)}(\mathbf{x}^1), \ldots, h_i^{(\ell)}(\mathbf{x}^P)\right)$, with a notation similar to the previous chapter. The key observation is that the distribution of pre-activations at a given layer ℓ, conditioned to the distribution of the activation at the previous layer $\ell - 1$ is a centered iid Gaussian:

$$p\left(\mathbf{h}_i^{(\ell+1)} \mid \mathbf{h}_i^{(\ell)}\right) = \frac{e^{\left(\mathbf{h}^{(\ell+1)}\right)^T [\hat{K}^{(\ell)}]^{-1}\mathbf{h}^{(\ell+1)}}}{\sqrt{2\pi \det \hat{K}^{(\ell)}}} \tag{2.7}$$

with experimental covariance $\hat{K}^{(\ell)}$:

$$\hat{K}^{(\ell)}_{\mu\nu} := \langle h_i^{(\ell+1)}(\mathbf{x}^\mu) h_i^{(\ell+1)}(\mathbf{x}^\nu)\rangle = \frac{1}{\lambda_b} + \frac{1}{\lambda_\ell N_\ell}\sum_{j=1}^{N_\ell} \sigma\left(h_j^{(\ell)}(\mathbf{x}^\mu)\right)\sigma\left(h_j^{(\ell)}(\mathbf{x}^\nu)\right). \tag{2.8}$$

By Levy's continuity theorem (see for instance [70]), we seek to show that:

$$\lim_{\{N_\ell\to\infty\}} \mathbb{E}\left[\exp\left(-i\sum_{i=1}^{N_L}\mathbf{h}_i^{(L+1)}\cdot\boldsymbol{\xi}_i\right)\right] = \exp\left(-\frac{1}{2}\sum_{i=1}^{N_L}\boldsymbol{\xi}_i^T K_X^{(L)}\boldsymbol{\xi}_j\right) \tag{2.9}$$

where

$$K_X^{(L)} = (K^{(L)}(\mathbf{x}^\mu,\mathbf{x}^\nu),\ (\mathbf{x}^\mu,\mathbf{x}^\nu)\in X), \tag{2.10}$$

and $K^{(L)}$ satisfyies Eq. (2.5). Using the property in (2.7), we can write:

$$p\left(\{\mathbf{h}_i^{(L+1)}\}\right) := \int D\mathbf{h}_i^{(L)} p\left(\mathbf{h}_i^{(L+1)} \mid \mathbf{h}_i^{(L)}\right) \tag{2.11}$$

$$= \int D\mathbf{h}_i^{(L)} \frac{e^{\sum_i \left(\mathbf{h}_i^{(L+1)}\right)^T [\hat{K}^{(L)}]^{-1}\mathbf{h}_i^{(L+1)}}}{\sqrt{2\pi\det\hat{K}^{(L)}}} \tag{2.12}$$

where I used again the short notation $D\mathbf{h} = p\left(\{\mathbf{h}_i\}\right)\prod_{i\mu}\mathrm{d}h_i^\mu$. Therefore:

$$\mathbb{E}\left[\exp\left(-i\sum_{i=1}^{N_L}\mathbf{h}_i^{(L+1)}\cdot\boldsymbol{\xi}_i\right)\right] \equiv \int \mathrm{d}\{\mathbf{h}_i^{(L+1)}\} P\left(\{\mathbf{h}_i^{(L+1)}\}\right) e^{-i\sum_{i=1}^{N_L}\mathbf{h}_i^{(L+1)}\cdot\boldsymbol{\xi}_i}$$

$$= \exp\left(\sum_{i=1}^{N_{L+1}}\boldsymbol{\xi}_i^T[\hat{K}_X^{(L)}]^{-1}\boldsymbol{\xi}_i\right) \tag{2.13}$$

We can rely on a Theorem for the structure of observables [1] to guarantee that the following converges in distribution:

$$\lim_{n\to\infty}\hat{K}^{(\ell)}(\mathbf{x}^\mu,\mathbf{x}^\nu) = K^{(\ell)}(\mathbf{x}^\mu,\mathbf{x}^\nu) := \lim_{n\to\infty}\mathbb{E}\left[\hat{K}^{(\ell)}(\mathbf{x}^\mu,\mathbf{x}^\nu)\right] \tag{2.14}$$

[1] whose proof and statement can be found in [71] Theorem 3.1 and Lemma 7.5.

with the result in Eq. (2.13), this immediately yields Eq. (2.9).

Thus, we see that $\left(h_i^{(L)}(\mathbf{x}^\mu), i = 1, \dots, N_L\right)$ indeed converges to independent zero-mean Gaussians with covariance $K_A^{(L)}$. Moreover, for any $\ell = 1, \dots, L$ we have:

$$\begin{aligned}
&\lim_{\{N_\ell \to \infty\}} \langle h_1^{(\ell+1)}(\mathbf{x}^\mu) h_1^{(\ell+1)}(\mathbf{x}^\nu) \rangle = \\
&\quad = \lim_{\{N_\ell \to \infty\}} \mathbb{E}\left[\hat{K}(\ell)(\mathbf{x}^\mu, \mathbf{x}^\nu)\right] \\
&\quad = \lim_{\{N_\ell \to \infty\}} \mathbb{E}\left[\frac{1}{\lambda_b} + \frac{1}{\lambda_\ell N^{(\ell)}} \sum_{j=1}^{n_\ell} \sigma\left(h_j^{(\ell)}(\mathbf{x}^\mu)\right) \sigma\left(h_j^{(\ell)}(\mathbf{x}^\nu)\right)\right] \\
&\quad = \frac{1}{\lambda_b} + \frac{1}{\lambda_\ell} \mathbb{E}_{K^{(\ell)}}\left[\sigma\left(h_1^{(\ell)}(\mathbf{x}^\mu)\right) \sigma\left(h_1^{(\ell)}(\mathbf{x}^\nu)\right)\right]
\end{aligned} \tag{2.15}$$

which confirms the recurrence relation in Eq. (2.5). □

2.1.2 Informal derivation in the physics formalism

The expression of the prior (and posterior) distribution of the outputs of a wide network can be re-derived informally using the physics formalism. This is instructive to introduce the reader to the method used in this work. For the sake of simplicity, consider a standard-scaled 1HL network with one scalar output. I recall its definition:

$$f_{\text{1HL}}(\mathbf{x}) = \frac{1}{\sqrt{N_1}} \sum_{i=1}^{N_1} v_i \, \sigma\left(\frac{W_i \cdot \mathbf{x}}{\sqrt{N_0}}\right) \tag{2.16}$$

where the weights are taken from a normal distribution with zero mean and unitary variance $v_i, W_{ij} \sim \mathcal{N}(0, 1)$. In the Bayesian setting that we are exploring, this scaling for the network's function and weights are equivalent to taking the standard NTK initialization (first row in table 1.1). With the usual notation for the networks ountputs $\mathbf{f} = (f(\mathbf{x}^\mu), \mathbf{x}^\mu \in \mathcal{D})$, we can write the joint probability distribution of the network outputs $p(\mathbf{f})$:

$$p(\mathbf{f}|X) = \int DW \, D\mathbf{v} \prod_\mu \delta\left(f^\mu - \frac{1}{\sqrt{N_1}} \sum_i v_i \, \sigma\left(\frac{W_i \cdot \mathbf{x}_\mu}{\sqrt{N_0}}\right)\right) \tag{2.17}$$

where the notation $DW \equiv p(\{W_{ij}\}) \prod_{ij} \mathrm{d}W_{ij}$ indicates integration over the elements with their joint probability distribution. For consistency of notation with the rest of the work, we shall denote the network's outputs and pre-activations respectively f^μ and $\mathbf{h}^\mu$. To proceed, we need to use the common identity:

$$1 = \int \prod_{i\mu} \mathrm{d}h_i^\mu \prod_{i\mu} \delta\left(h_i^\mu - \frac{W_i \cdot \mathbf{x}_\mu}{\sqrt{N_0}}\right) \tag{2.18}$$

Rewriting the previous expression with the deltas in their standard exponential representation one gets:

$$\begin{aligned} p\left(\mathbf{f}|X\right) = \int \prod_{\mu} \mathrm{d}\bar{f}^\mu \prod_{i\mu} \mathrm{d}h_i^\mu \, e^{i\sum_\mu \bar{f}^\mu f^\mu + i\sum_{i\mu} \bar{h}_i^\mu h_i^\mu} \times \\ \times \int DW \, D\mathbf{v} \, e^{-i\sum_\mu \frac{\bar{f}_\mu}{\sqrt{N_1}} \sum_i v_i \, \sigma\left(h_i^\mu\right) - i\sum_{i\mu} \bar{h}_i^\mu \frac{W_i \cdot \mathbf{x}_\mu}{\sqrt{N_0}}} \end{aligned} \tag{2.19}$$

The index i can be factorized, given that the probability distributions of the weights are factorized over the hidden layer neurons, and the remaining Gaussian integrals can be performed:

$$\begin{aligned} p\left(\mathbf{f}|X\right) = \int \prod_{\mu} \mathrm{d}\bar{f}^\mu \; e^{i\sum_\mu \bar{f}^\mu f^\mu} \times \\ \times \left[\int \prod_{\mu} \mathrm{d}h^\mu \, \mathrm{d}\bar{h}^\mu e^{i\sum_\mu \bar{h}^\mu h^\mu - \frac{1}{N_1}\left(\sum_\mu \bar{f}_\mu \sigma(h^\mu)\right)^2 - \frac{1}{2}\sum_{\mu\nu} \bar{h}^\mu C_{\mu\nu} \bar{h}^\nu}\right]^{N_1} \end{aligned} \tag{2.20}$$

where we have used the notation C for the correlation matrix of the dataset C, which has elements $C_{\mu\nu} \equiv \frac{\mathbf{x}^\mu \cdot \mathbf{x}^\nu}{N_0}$. The $\bar{h}^\mu$ integrals are Gaussian and can be performed:

$$p\left(\mathbf{f}|X\right) = \int \prod_{\mu} \mathrm{d}\bar{f}^\mu \; e^{i\sum_\mu \bar{f}^\mu f^\mu} \left[\int \frac{\prod_\mu \mathrm{d}h^\mu}{\sqrt{\mathrm{det}C}} e^{-\frac{1}{2N_1}\left(\sum_\mu \bar{f}_\mu \sigma(h^\mu)\right)^2 - \frac{1}{2}\sum_{\mu\nu} h^\mu C^{-1}_{\mu\nu} h^\nu}\right]^{N_1} \tag{2.21}$$

Note that I didn't use any approximation to obtain the previous expression, and indeed this is the last exact expression that we can obtain for the prior over functions of a 1HL network. The expression unsurprisingly simplifies in the infinite-width limit: if one takes the limit of large width $N_1 \to \infty$, keeping the size of the dataset P fixed, the exponential term in the integral

can be expanded in Taylor series around 1:

$$
\begin{aligned}
&p\left(\mathbf{f}|X\right) = \\
&= \int \prod_{\mu} \mathrm{d}\bar{f}^{\mu}\, e^{i\sum_{\mu}\bar{f}^{\mu}f^{\mu}} \left[1 - \frac{1}{2N_1}\int \prod_{\mu} \mathrm{d}h^{\mu}\, \mathcal{N}_h(0,C) \sum_{\mu\nu} \bar{f}_{\mu}\sigma\left(h^{\mu}\right)\sigma\left(h^{\nu}\right)\bar{f}_{\nu}\right]^{N_1} \\
&\underset{N_1\to\infty}{\to} \int \prod_{\mu} \mathrm{d}\bar{f}^{\mu}\; e^{i\sum_{\mu}\bar{f}^{\mu}f^{\mu}} \left[1 - \frac{1}{2N_1}\sum_{\mu\nu} \bar{f}_{\mu}K_{\mu\nu}\bar{f}_{\nu}\right]^{N_1} = \mathcal{N}_f(0,K)
\end{aligned}
\tag{2.22}
$$

where the kernel matrix with elements $K_{\mu\nu}$ is the same defined in Eq. (1.42). This result is equivalent to the one found in [27]. It is stressing that the NNGP kernel we find here differs from the neural tangent kernel (NTK) that is found in the infinite-width limit of networks trained under gradient descent [72], that I will breefly discuss in Section 2.2. The fact that the infinite-width limit of a Bayesian neural network differs from the one obtained from gradient descent is indeed known and discussed in literature [32].

Posterior over outputs

From the physical point of view, we seek to compute the following partition function

$$
Z = \int e^{-\frac{\beta}{2}\|\mathbf{f}-\mathbf{y}\|^2} p(\mathbf{f}|X)\,\mathrm{d}\mathbf{f} \tag{2.23}
$$

This integral is computed exactly from the probability distribution in (2.22), and yields:

$$
Z = \mathcal{N}_{\mathbf{y}}\left(0, K + \frac{\mathbb{1}}{\beta}\right) \tag{2.24}
$$

As we have seen in Section 1.3.2, the bayesian posterior distribution can be computed from the joint probability distribution of the outputs of the trainset $\mathbf{f}$ and the output f^* corresponding to a new example (or set of examples) $\mathbf{x}^*$, that is $p\left(f^*, \mathbf{f}|\mathbf{x}^*, X\right)$. This is precisely the integral in Eq. (1.50), and corresponds to the generic Bayesian posterior at temperature $1/\beta$. In the statistical mechanics formalism, having chosen the MSE loss in Eq. 1.18, we directly compute the generalization error, defined in (1.32), that is the average quadratic deviation of the new

output from the desired one:

$$\langle \epsilon_g(\mathbf{x}^*, y^*) \rangle = (y^* - \Gamma) + \Sigma \tag{2.25}$$

with Γ and Σ defined in (1.50).

2.1.3 Infinite-width limit of neural networks with local connections

Whereas the original work on the infinite-width limit considered deep architectures with FC hidden layers only, the generalization to architectures with convolutional layers is straightforward if one replaces neurons in the FC hidden layers with the channels/convolutional filters of the CNN, that is $N_c \gg P$. This result was firstly obtained in [29], and a straightforward derivation follows from computations in Appendix E. Consider for simplicity a 1HL NN with 1d convolutional filters in the first layer plus a readout layer:

$$\begin{aligned} f_{\text{CNN}}(\mathbf{x}) &= \sum_{i=1}^{\lfloor N_0/S \rfloor} \sum_{a=1}^{N_c} v_i^a \sigma\left[h_i^a(\mathbf{x})\right] \,, \\ h_i^a(\mathbf{x}) &= \sum_{m=-\lfloor M/2 \rfloor}^{\lfloor M/2 \rfloor} W_m^a x_{Si+m} \,. \end{aligned} \tag{2.26}$$

In the limit of infinitely many channels $N_c \gg P$, studying the standard NTK parameterization:

$$W_{ij} \sim \mathcal{N}\left(0, \frac{1}{\lambda_0 \, M}\right) \qquad v_i \sim \mathcal{N}\left(0, \frac{1}{\lambda_1 \, N_c \lfloor N_0/S \rfloor}\right) \qquad b^\ell \sim \mathcal{N}\left(0, \frac{1}{\lambda_b}\right) \tag{2.27}$$

one finds that the dynamics of the network is governed by a fixed averaged kernel $\bar{K}^L_{\mu\nu}$:

$$\bar{K}_{\mu\nu} = \frac{1}{\lambda_1 \lfloor N_0/S \rfloor} \sum_{i=1}^{\lfloor N_0/S \rfloor} K^{ii}_{\mu\nu} \,. \tag{2.28}$$

that is expressed in terms of a *local* kernel $K^{ij}_{\mu\nu}$:

$$K^{ij}_{\mu\nu} = \int d^2t\, \mathcal{N}_t\left(0, \tilde{C}^{ij}_{\mu\nu}\right) \sigma(t_1)\sigma(t_2)\,, \qquad \tilde{C}^{ij}_{\mu\nu} = \begin{pmatrix} C^{ii}_{\mu\mu} & C^{ij}_{\mu\nu} \\ C^{ij}_{\mu\nu} & C^{jj}_{\nu\nu}\,. \end{pmatrix}\,. \tag{2.29}$$

The indices i, j run over the local patches of the input parsed by the convolutional filters. Precisely, the local covariance matrix $C^{ij}_{\mu\nu}$ is given by:

$$C^{ij}_{\mu\nu} = \frac{1}{\lambda_0 M} \sum_{m=-\lfloor M/2 \rfloor}^{\lfloor M/2 \rfloor} x^{\mu}_{Si+m} x^{\nu}_{Sj+m}, \tag{2.30}$$

This quantity provides information on the self-correlation of the local patch in position Si for any pair of training patterns μ, ν in the dataset. In the same paper [29],the authors show that similar considerations hold also for deeper networks with finite arbitrary depth L. The deep CNN-GP is given in terms of the last layer Kernel $\bar{K}^L$, that satisfies a recursion relation similar to (2.7).

Interestingly, if one considers networks with local connections but without weight sharing, that is locally connected networks (LCNs), the infinite-width description that arised contains the same GP kernel as the convolutional neural networks. The two straightforward observations that I am going to investigate more in detail in chapter 5. (i) the infinite-width description is not sufficient to describe the effect of weight sharing, (ii) both LCNs and CNNs use only the information contained in the *diagonal* elements of the local kernel matrix $K^{ij}_{\mu\nu}$, since the elements with $i \neq j$ do not to partecipate to making predictions on unseen examples.

2.2 Neural tangent kernel: the infinite-width limit under gradient descent

In the previous sections I discussed the kernel regime that arises in the Bayesian setting for wide networks, here I will show that the same regime (with a different kernel) emerges for

a different training dynamics, that is gradient descent (GD). The fixed kernel that governs the network dynamics under GD is known as neural tangent kernel (NTK), first obtained in 2018 [30]. Since then, an influential line of work study the NTK or linear regime of neural networks [32, 73, 74]. The study of networks under GD departs from the scopes of the present work, therefore I will report the formal statement for the sake of completeness, following the pedagogical exposition in [69].

Consider the differential equation associated with the GD dynamics:

$$\frac{\mathrm{d}\theta_t}{\mathrm{d}t} = -\nabla_\theta \mathcal{L}(\theta_t). \tag{2.31}$$

Taking the Loss function to be the mean squared error defined in Eq. (1.18), the gradient is easily computed:

$$\nabla_\theta \mathcal{L}(\theta) = \sum_{\mu=1}^{P} \left(f(\mathbf{x}^\mu - y^\mu)\right) \nabla_\theta f_\theta(\mathbf{x}^\mu) \tag{2.32}$$

If one wants to look at the evolution of the function under gradient descent:

$$\frac{\mathrm{d}f(\mathbf{x})}{\mathrm{d}t} = \frac{\mathrm{d}f_\theta(\mathbf{x};\theta)}{\mathrm{d}\theta}\frac{\mathrm{d}\theta}{\mathrm{d}t} = -\eta \sum_{\mu=1}^{P} (f(\mathbf{x}^\mu, \theta) - y^\mu) \nabla_\theta f(\mathbf{x}^\mu;\theta) \nabla_\theta f(\mathbf{x};\theta) \tag{2.33}$$

We can isolate the so-called neural tangent kernel (NTK) $\Theta(\mathbf{x}, \mathbf{x}')$:

$$\Theta(\mathbf{x}, \mathbf{x}', \theta) = \nabla_\theta f(\mathbf{x}, \theta) \nabla_\theta f(\mathbf{x}', \theta). \tag{2.34}$$

Using the usual notation $\mathbf{f}(\theta) = (f(\mathbf{x}^\mu),\ \mathbf{x}^\mu \in X)$ for the vector of network's outputs, where I explicited the θ-dependence, we can write:

$$\mathbf{f}(\theta_{t+1}) \sim \mathbf{f}(\theta_t) - \eta \Theta(\theta_t) \nabla_f \mathcal{L}(f) \tag{2.35}$$

where $\Theta(\theta) = (\Theta(\mathbf{x}^\mu \mathbf{x}^\mu, \theta),\ \mathbf{x}^\mu, \mathbf{x}^\nu \in X)$ is the collection of the NTKs evaluated on all the possible pairs of trainset elements.

Linear dynamics

The main result of [30] is the following Theorem, it can be equivalently found in [73–75].

Theorem 2 (Linear regime of neural networks under GD). *At fixed length L, fix a function $\sigma : \mathbb{R} \to \mathbb{R}$ with $\sigma \in C^2$ polynomially bounded. Using the short notation θ for the ensemble of the network parameters, with initial standard condition θ_0 as in Eq.* (2.1)*. If $N_1, \dots, N_L = poly(P)$ then, with high probability:*

1. *The dynamics will converge to a perfectly interpolating solution:*

$$\lim_{t\to\infty} \mathcal{L}(\theta_t) = 0. \tag{2.36}$$

2. *The network stays close to its linearization around $t = 0$ at every optimization step t, meaning that there exists $\alpha > 0$ such that*

$$\sup_{t\geq 0} \|\theta_t - \theta_t^{\text{lin}}\| = O\left(\min\{N_1, \dots, N_L\}^{-\alpha}\right) \tag{2.37}$$

where

$$\frac{\mathrm{d}\theta_t^{\text{lin}}}{\mathrm{d}t} = -\nabla_\theta \mathcal{L}^{\text{lin}}(\theta_t^{\text{lin}}), \qquad \mathcal{L}_{lin}(\theta) = \sum_{\mu=1}^{P} \left(f_{\text{FCN}}^{\text{lin}}(\mathbf{x};\theta) - y_i\right)^2, \tag{2.38}$$

$$f_{\text{FCN}}^{\text{lin}}(\mathbf{x};\theta) = f_{\text{FCN}}(\mathbf{x};\theta_0) + \nabla_\theta f_{\text{FCN}}(\mathbf{x};\theta_0) \cdot (\theta - \theta_0). \tag{2.39}$$

A wide network that is trained under GD is said to be in a regime of *lazy training*, in the sense that its weights never move much away from the initial conditions θ_0, so that the dynamics can be thought as a taylor expansion of the networks around θ_0.

CHAPTER 3

Understanding feature learning in deep neural networks: an open challenge

Neural networks have long been known to perform a kind of feature extraction from the training data [4, 37, 39], and this is also widely believed to be at the basis of their success. This concept, often called *feature learning*, is not technically well defined, despite extensive literature that highlights the phenomenon [37–39, 76, 77]. The reader will not be surprised to discover that there is an effort by theoreticians to try and put it on theoretical grounds. Despite some preliminary attempts to give a general definition of feature learning [11, 38, 78–86], this phrasing has progressively taken the connotation of "far from the kernel limit", or equivalently, far from the infinite-width regime.

In this Chapter I will present a series of recent experimental results that point in the direction that, whatever form of feature learning occurs in FC networks, is not extremely effective in terms of improving generalization performance, even at finite width. On the other hand, CNNs seem to be able to extract useful representations from train data even in a heavily overparameterized regime of learning (that I will describe in Chapter 4), as testified by the experimental advantage of employing a small number of convolutional filters. In the final

Chapter of this work, I will try and rationalize the aforementioned observation effectively quantifying, through the mechanism of *local kernel renormalization*, what it means to be "far" from the kernel regime, providing a possible description of what it means to learn features in these models.

3.1 Limitations of the infinite-width limit of standard scaled networks

Although fundamental theoretical progress has been achieved in the infinite-width limit, this regime is known to fail in grasping certain salient aspects of learning in neural networks. Two remarkable examples are listed below.

1. **There is no feature learning in infinitely wide networks with the standard scaling.** The stochastic process that describes information flow in infinitely-wide deep neural network is a familiar Gaussian process, which is completely determined by a non-linear kernel. A fundamental consequence of this finding is that learning in the infinite-width limit is equivalent to kernel learning [8, 20, 33, 34] with a static kernel whose form depends on the learning algorithm, completely fixed by the statistics of the weights at initialization, that does not evolve during training. Since the kernel is the quantity that incorporates all the inormation on the structure of the data, this last observation suggests that feature learning is essentially absent in infinite-width networks [42, 87]. This consideration is strictly linked to the choice of the standard scaling, it is indeed worth mentioning that different claims of existence of feature learning in the infinite-width limit exist in the so-called mean field scaling, see for instance [11, 86].

2. **Infinite-width networks are not affected by weight sharing.** The infinite-width averaged Kernel (2.28) is the same for all the architectures that have local connections between weights, independently from the fact that those weights are shared among the local patches or not. The success of CNNs, however, is indeed linked to weight sharing,

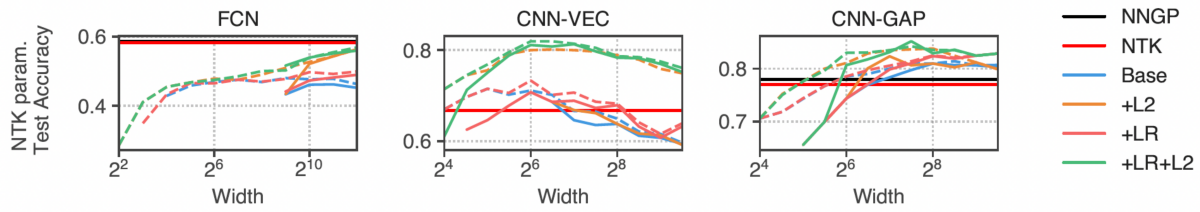


Figure 3.1 –Test accuracy as a funcion of network's width. In FCNs, the experimental test accuracy of finite-width networks approaches its asymtptotic performance (NNGP kernel performance) from below. FCNs don't outpeform their infinite-width GP. CNNs on the other hand, do best their infinite-width kernel, both with and without average pooling layer (GAP). Reprinted from [72]

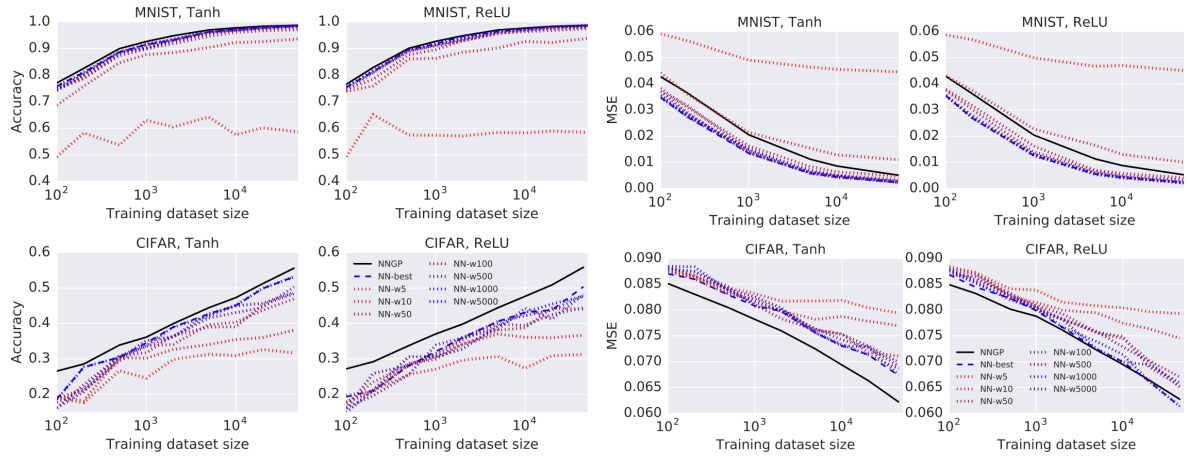
as is testified by the better performances in terms of generalization of CNNs compared to LCNs [29, 88]. The infinite-width analysis is therefore insufficient to describe the effect of sharing parameters in a NN.

3.2 Empirical evidence for feature learning at finite width: fully-connected VS locally connected

To the end of understanding feature learning regimes and its difference from the infinite-width behaviour, plenty of recent literature aims at numerically compare performances of finite-width networks with different geometries (FC layers, convolutions, pooling, local connections, etc) and their correspondant GP-kernels [29, 32, 72, 89].

A notable example is the large-scale empirical study conducted by the authors of Ref. [72], where they compare finite-width FC networks, CNNs with finite number of channels and their infinite-width limit kernel counterparts. A variety of possible settings to improve the generalization performance is explored combining centering, large learning rate, L2 regularization, ensembling, underfitting by early-stopping and ZCA whitening of the inputs. This massive analysis provides a few striking empirical observations highlited in Figure 3.1:

(i) Infinite-width kernels systematically outperform their finite-width counterpart in the case of FC-DNNs;



(a) Accuracy (b) Mean squared error

Figure 3.2 –Comparison between performances of FC finite-width fully trained networks (dotted lines) with $tanh$ activation and correspondant infinitely wide bayesian networks (black solid line). The authors compare generalization performances measuring both the test accuracy (left) and mean squared error (MSE) (right) on two benchmark computer vision datasets, MNIST (above) and CIFAR (below). Except for the MSE for MNIST, the NNGP always outperforms the finite-width network for every dataset size in a two-decades range. Figure reprinted from [27]

(ii) CNNs with finite number of channels often outperform their corresponding infinite-width kernel performance.

Such empirical evidence is further corroborated by a striking difference in the behavior of the test accuracy as a function of the size of the HLs/number of channels in FC/CNNs: whereas for FC networks the accuracy monotonically approaches the kernel performance from below, in CNNs one often observes a non-monotonic behavior and a non-trivial region at finite-width where the network is capable of outperforming the CNN GP performance. Several similar observations have been reported in the literature: the authors of Ref. [89] very recently pointed out that infinite-width deep FC neural networks eventually outperform their finite-width counterpart as the size of the training set grows. In one of the seminal papers dealing with the infinite-width limit [27], the authors observe that increasing the hidden layers size almost always leads to optimal test accuracy on deep FC architectures trained on MNIST and on CIFAR10, two of the benchmark datasets for computer vision learning problems (see Figure 3.2).

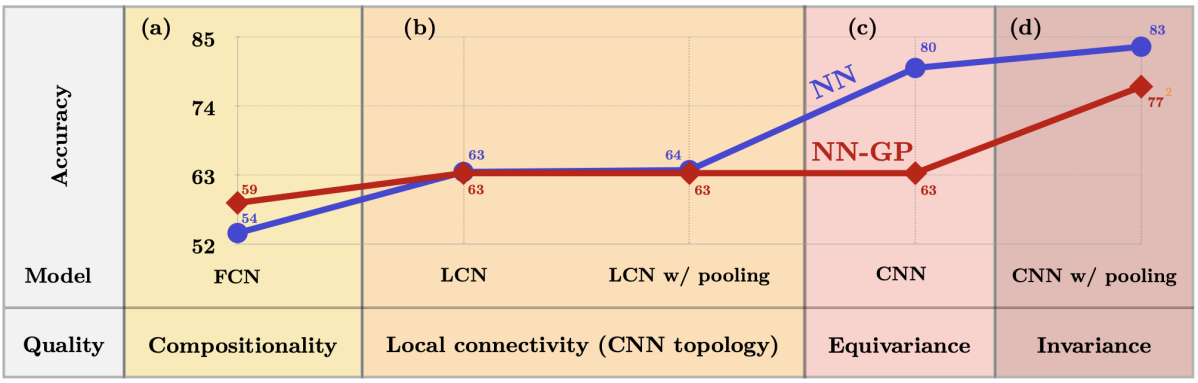


Figure 3.3 –Comparison between performances of SGD trained (blue) and infinitely wide bayesian networks (red). From left to right: (i) when only FC layers are employed, infinite-width kernel outperforms finite-width trained network. (ii) When only local interaction is inserted in the model (see Eq.), finite and infinite-width networks performs identically; (iii) as soon as convolutions are employed, trained finite-width networks systematically outperform their infinite-width GP, wheter or not pooling layer is addedd. Figure reprinted from [29]

Another notable analysis is found in Ref. [29], where the authors also test networks with *local connections* but without weight sharing (LCNs). A brief summary of their results is reported in Figure 3.3. The points discussed above suggest that deep FC architectures and LCNs are not as effective at finite-width as their correspondant GP kernels, indicating that in these regimes, feature learning is essentialy absent. This is in sharp contrast to what practitioners observe using state-of-the-art deep networks with convolutional layers, which achieve optimal generalization performance at finite width when feature learning can occur and is in fact essential. This mismatch triggers at least two conceptual questions:

(i) Why is it ultimately convenient to employ large-width architectures when only FC layers are available?

(ii) Why is this not the case when convolutional layers are employed? And how does a CNN operatively exploit the finite-width regime for efficient feature learning?

Preliminary theoretical work in the direction of understanding the feature learning regime of deep nets was carried out in Refs. [11, 78–86]. The authors of Refs. [90, 91] analytically investigated the advantages of employing convolutional neural tangent kernels in the infinite-width limit and understood why infinite-width FCNs perform worse in the mean field regime

than in the lazy-training one [92]. The interplay between the lazy training regime and the mean field limit [64] has been the subject of a thorough investigation in [93, 94].

In the following Chapter, I will introduce the reader to a framework in which the aforementioned empirical observations can be partly rationalized through the lens of *kernel renormalization*. This theoretical framework, firstly developed in [95], provides a simple, yet very instructive answer to question (i) for 1HL networks. It is in fact possible to show that the performance of a finite-width shallow network can be obtained with the corresponding infinite-width kernel and a suitable choice of the Gaussian priors over the weights of each layer. Moreover, as I will discuss in detail in Chapter 5, these findings can be employed to provide preliminary insight to question (ii).

Part II

Results: A statistical mechanics framework for deep neural networks beyond the infinite-width limit

CHAPTER 4

One way to go beyond the infinite-width limit: the proportional regime

The infinite-width limit of neural networks has been mostly understood in deep feedforward architectures with several geometries [26, 27, 29, 63], and both for Bayesian and GD-trained networks [30, 35]. The limitations that an infinite-width description carries, as we have seen in the previous Chapter, suggest that this description is falining to grasp fundamental aspects of realistic machine learning, making it necessary from the theoretical point of view to go *beyond* it.

Here and in the following, what I will mean by going beyond the infinite-width is to explore a less-overparameterized and more realistic scaling of the parameters with respect to the number of avaliable datapoints, in a data-dependent setting. Specifically, I will consider a *proportional regime*, where one lets the size of the training set P scales to infinity at the same rate as the

size of the hidden layers N_ℓ:

$$P, N_\ell \to \infty, \quad \alpha_\ell = \frac{P}{N_\ell} \text{ finite} \qquad \forall \ell = 1, \dots, L \tag{4.1}$$

with L being the (finite) depth of the network. This scaling choice guarantees that such networks still work in the overparametrized regime, like the infinite-width ones, but with constraints that are more realistic compared to practical learning scenarios. The scaling of P with the input size N_0, which deserves special care, is discussed below in Sec. A.2. It is worth remarking that a statistical mechanics description of networks in the proportional regime can also be found in [9, 96], for toy *linear* architectures, or in a teacher-student scenario and averaging over Gaussian data distribution in [23, 97]. Note that the proportional regime is just one possilbe choice to move from the infinite-width description. To mention a couple of relevant strategies, one could for instance study different scalings from the NKT, as was done in [11, 78], or consider the limit of infinitely deep networks $L \to \infty$ [98].

Supervised learning problem

We consider a supervised learning problem with training set $\mathcal{D}_P = \{\mathbf{x}^\mu, y^\mu\}_{\mu=1}^P$, where each $\mathbf{x}^\mu \in \mathbb{R}^{N_0}$ and the corresponding labels $y^\mu \in \mathbb{R}$. We analyse regression problems with a regularized quadratic loss function:

$$\mathcal{L}(\theta) = \mathcal{L}_{\text{MSE}}(\theta) + \mathcal{L}_{\text{REG}}(\theta) \tag{4.2}$$

$$\mathcal{L}_{\text{MSE}}(\theta) = \frac{1}{2} \sum_{\mu=1}^{P} \left[y^\mu - f_{\text{DNN}}(\boldsymbol{x}^\mu) \right]^2 , \tag{4.3}$$

$$\mathcal{L}_{\text{REG}}(\theta) = \frac{1}{2\beta} \sum_{\ell=1}^{L} \lambda_\ell \, \|W^{(\ell)}\|^2 + \frac{\lambda_L}{2\beta} \|\mathbf{v}\|^2 \tag{4.4}$$

where $\beta = 1/T$ is the inverse temperature, and $\|\cdot\|$ is the standard Frobenius norm defined for the weights matrices $W^{(\ell)}$. Scaling $\mathcal{L}_{\text{reg}}$ by $1/\beta$ has a natural Bayesian learning interpretation: the Gibbs probability $P_\beta(\theta) = Z^{-1} e^{-\beta \mathcal{L}(\theta)}$ associated with the partition function in equation (1.29) is the posterior distribution of the weights after training.

Fully connected NNs with the standard scaling in the Bayesian setting

Throughout this Chapter, I will consider deep neural networks $f_{\text{DNN}}(\mathbf{x})$ with $(L-1)$ fully-connected hidden (FC) layers and a final linear readout layer, defined as follows:

$$
\begin{aligned}
f_{\text{FCN}}(\mathbf{x}) &= \frac{1}{\sqrt{\lambda_L N_L}} \sum_{i_L=1}^{N_L} v_{i_L} \sigma\left[h_{i_L}^{(L)}(\mathbf{x})\right] + b_{i_L} \,, \\
h_{i_\ell}^{(\ell)}(\mathbf{x}) &= \frac{1}{\sqrt{\lambda_\ell N_\ell}} \sum_{i_{\ell-1}=1}^{N_{\ell-1}} W_{i_\ell i_{\ell-1}}^{(\ell)} \sigma\left(h_{i_{\ell-1}}^{(\ell-1)}(\mathbf{x})\right) + b_{i_\ell}^{(\ell)} \,, \qquad \ell = 2 \dots L-1 \\
h_{i_1}^{(1)}(\mathbf{x}) &= \frac{1}{\sqrt{\lambda_0 N_0}} \sum_{i_0=1}^{N_0} W_{i_1 i_0}^{(1)} x_{i_0} + b_{i_1}^{(1)}
\end{aligned}
\tag{4.5}
$$

Defining the network in this way, I can choose all the weights from a standard normal distribution $\mathcal{N}(0,1)$. In the framework where one is interested in computing the partition function in (1.29), this is equivalent to scaling the weights with the NTK scaling. In other words, standard and NTK parameterizations are equivalent in this setting (rows one and two in Table 1.1 are equivalent).

4.1 Effective action of Bayesian shallow fully-connected neural networks

In the case of 1HL architectures, which implement the function in Eq (4.5) with $L = 1$, the partition function (1.29) can be reduced to a two-variable integral, in the thermodynamic limit described in Eq. (4.1). This is the first and main result of a recent paper [95], and of the present work. The partition function, expressed in terms of an *effective action* S, reads:

$$
Z = \int \mathrm{d}Q \int \mathrm{d}\bar{Q} \, \exp\left[-\frac{N_1}{2} S(Q, \bar{Q})\right] \,. \tag{4.6}
$$

where the action S is given by:

$$S = -Q\bar{Q} + \log(1+Q) + \frac{\alpha_1}{P}\,\mathrm{Tr}\log\beta\left[\frac{\mathbb{1}}{\beta} + K^{(\mathrm{R})}(\bar{Q})\right] + \\ + \frac{\alpha_1}{P}\mathbf{y}^\top\left[\frac{\mathbb{1}}{\beta} + K^{(\mathrm{R})}(\bar{Q})\right]^{-1}\mathbf{y}\,, \tag{4.7}$$

with

$$K^{(\mathrm{R})}(\bar{Q}) = \frac{\bar{Q}}{\lambda_1}K(C)\,, \qquad C_{\mu\nu} = \frac{1}{\lambda_0}\frac{x^\mu\cdot x^\nu}{N_0}\,. \tag{4.8}$$

The input-dependent kernel $K(C)$ is the neural network Gaussian process (NNGP) kernel [27] arising in the infinite-width limit (1.43). Remarkably, the *renormalized* kernel $K^{(\mathrm{R})}$ that enters the effective action is the NNGP kernel rescaled by the parameter $\bar{Q}$. At the saddle point, one can easily express Q as a function of $\bar{Q}$, leaving the latter as the only free parameter that a standard-scaled proportional-width network can optimize during training.

Minimizing the effective action is straightforward if $\alpha_1 \to 0$, since one easily finds that $Q^* = 0$, $\bar{Q}^* = 1$ and recovers the well-known infinite-width limit NNGP kernel. At finite α_1 one finds data-dependent solutions for the order parameter that produce a global renormalization of the infinite-width kernel as expressed by Eq. (4.8). In other words, the difference between learning with standard-scaled networks in the infinite-width and proportional-width regime amounts to the optimization of the free parameter $\bar{Q}$. A sketch of derivation of Eq. (4.7) can be found in the next subsection, while details of the calculations are in Appendix B.2. For many reasonable non-linearities and input data distributions the derivation goes through at least in the regime $P = O(N_0)$ (see A.2). In [95], we conjecture that this result is exact since the only key Gaussian approximation that we perform is justified by the Breuer-Major theorem [99], as shown in Appendix A.

Sketch of the derivation

In this section I will discuss the salient aspects of the calculation of the 1HL partition function in Eq. (4.6). The starting point is the definition in Eq. (2.23), that can be rewritten as:

$$Z = \int \prod_{i_1}^{N_1} \mathrm{d}v_{i_1} \prod_{i_1,i_0}^{N_1,N_0} \mathrm{d}w_{i_1 i_0} \, \exp\Bigg\{ -\frac{\lambda_1}{2} \sum_{i_1}^{N_1} v_{i_1}^2 - \frac{\lambda_0}{2} \|w\|^2 \\ - \frac{\beta}{2} \sum_{\mu}^{P} \left[y^\mu - \frac{1}{\sqrt{N_1}} \sum_{i_1}^{N_1} v_{i_1} \sigma\left(\sum_{i_0}^{N_0} \frac{w_{i_1,i_0} x_{i_0}^\mu}{\sqrt{N_0}} \right) \right]^2 \Bigg\}, \quad (4.9)$$

where w, v are respectively the first and last layer weights, and σ is the non-linear activation function. This computation holds for zero-mean activation functions, that is functions whose average over a centered Gaussian is zero. This restriction can be lifted, as I show in the Appendix, Sec. C.3, deriving an effective action for the generic finite-mean case.

The first step is to decouple the weights of the different layers in the loss function. This can be done including standard identities built over two families of Dirac deltas, one for the pre-activations of the hidden layer and one for the output of the network:

$$1 = \int \prod_{\mu}^{P} ds^\mu \delta \left[s^\mu - \frac{1}{\sqrt{N_1}} \sum_{i}^{N_1} v_{i_1} \sigma(h_{i_1}^\mu) \right] , \quad (4.10)$$

$$1 = \int \prod_{\mu}^{P} \prod_{i_1}^{N_1} dh_{i_1}^\mu \, \delta \left(h_{i_1}^\mu - \frac{1}{\sqrt{N_0}} \sum_{i_0}^{N_0} w_{i_1 i_0} x_{i_0}^\mu \right) . \quad (4.11)$$

By using a standard Fourier representation of these deltas, which introduces the conjugate variables $\bar{h}_{i_1}^\mu$ and $\bar{s}^\mu$, we can perform the Gaussian integrals on the internal and external weights. Using the fact that the $h_{i_1}^\mu, \bar{h}_{i_1}^\mu$ integrals can be factorized over the index i_1, and performing the former Gaussian integration one has:

$$Z = \int \prod_{\mu}^{P} \frac{\mathrm{d}s^\mu \, \mathrm{d}\bar{s}^\mu}{2\pi} e^{-\frac{\beta}{2} \sum_\mu (y^\mu - s^\mu)^2 + i \sum_\mu^P s^\mu \bar{s}^\mu} \left[\int \prod_{\mu}^{P} \frac{\mathrm{d}h^\mu}{2\pi} e^{-\frac{1}{2\lambda_1 N_1} \left[\sum_\mu^P \bar{s}^\mu \sigma(h^\mu) \right]^2} P_1(\{h^\mu\}) \right]^{N_1} , \quad (4.12)$$

where

$$P_1(\{h^\mu\}) = \frac{e^{-\frac{1}{2}\sum_{\mu,\nu}^{P} h^\mu \left[C^{-1}\right]_{\mu\nu} h^\mu}}{\sqrt{(2\pi)^P \det C}} \,, \; C_{\mu\nu} = \frac{1}{\lambda_0 N_0} \sum_{i_0}^{N_0} x_{i_0}^\mu x_{i_0}^\nu \,. \tag{4.13}$$

To deal with the integral over h^μ we can include a further Dirac delta identity for the random variable $q = 1/\sqrt{\lambda_1 N_1} \sum_\mu \bar{s}^\mu \sigma\left(h^\mu\right)$. This leaves us with the problem of finding the probability density $P(q)$. In the limit defined in (4.1), this is exactly the same setting of the Breuer-Major theorems [99, 113, 114]. As such, it is sufficient that both the (regularized) covariance C and the activation function σ satisfy the hypotheses of the theorem to guarantee that the probability distribution $P(q)$ converges in distribution to a Gaussian:

$$\begin{aligned} P(q) &= \int \mathrm{d}^P h \, P_1(\{h^\mu\}) \delta\left[q - \frac{1}{\sqrt{\lambda_1 N_1}} \sum_\mu \bar{s}^\mu \sigma\left(h^\mu\right)\right] \to \mathcal{N}_q(0, Q(\bar{s})) \,, \\ Q(\bar{s}, C) &= \frac{1}{\lambda_1 N_1} \sum_{\mu,\nu}^{P} \bar{s}^\mu K_{\mu\nu}(C) \bar{s}^\nu \,. \end{aligned} \tag{4.14}$$

where the matrix $K(C)$ is the NNGP kernel defined in (1.43). Now we can integrate over the variable q and obtain:

$$\left[\int \frac{\mathrm{d}q \, e^{-\frac{q^2}{2} - \frac{q^2}{2Q(\bar{s},C)}}}{\sqrt{2\pi Q(\bar{s}, C)}}\right]^{\frac{N_1}{2}} = \left[Q(\bar{s}, C) + 1\right]^{-\frac{N_1}{2}} \,. \tag{4.15}$$

In the general case of finite $\alpha_1 = P/N_1$, we are only left with the integrals in s^μ and $\bar{s}^\mu$. To solve them it is convenient to introduce one final Dirac delta identity:

$$1 = \int dQ \, \delta\left[Q - \frac{1}{\lambda_1 N_1} \sum_{\mu,\nu} \bar{s}^\mu K(C)_{\mu\nu} \bar{s}^\nu\right] . \tag{4.16}$$

Finally, the integrals in s^μ and $\bar{s}^\mu$ are Gaussian once another integral representation of the delta via a conjugate variable $\bar{Q}$ is inserted. This allows us to get the final effective action obtained in equation (4.7).

Generalization error

A computation similar to the one presented above shows that the generalization error, defined in Eq. (1.32), can be expressed in terms of the usual bias-variance decomposition:

$$\epsilon_g(\mathbf{x}^0, y^0) = (y_0 - \Gamma)^2 + \sigma^2 \tag{4.17}$$

$$\Gamma_1 = \frac{\bar{Q}}{\lambda_1} \sum_{\mu\nu} \kappa_\mu(\mathbf{x}^0) \left(\frac{\mathbb{1}}{\beta} + \frac{\bar{Q}}{\lambda_1} K \right)^{-1}_{\mu\nu} y_\nu,$$

$$\sigma_1^2 = \frac{\bar{Q}}{\lambda_1} \left[\kappa_0(\mathbf{x}^0) - \frac{\bar{Q}}{\lambda_1} \sum_{\mu\nu} \kappa_\mu(\mathbf{x}^0) \left(\frac{\mathbb{1}}{\beta} + \frac{\bar{Q}}{\lambda_1} K \right)^{-1}_{\mu\nu} \kappa_\nu(\mathbf{x}^0) \right] \tag{4.18}$$

where $K = K(C)$ is the NNGP kernel defined in (1.43), $\kappa_\mu(\mathbf{x}^0) \equiv K(\mathbf{x}^0, \mathbf{x}^\mu)$, with the NNGP kernel function defined in (1.42), and $\kappa_0(\mathbf{x}^0) \equiv K(\mathbf{x}^0, \mathbf{x}^0)$. See Appendix B for further details on how to analytically compute the generalization error, and Chapter 5 for numerical tests of this result.

Additional results

In the supplemental material, I reported a number of additional results that partially represent ongoing work: (i) a re-derivation of the effective action in the case of linear activation function, valid at fixed P, N_1, N_0, together with a comparison with the results given in [9, 98]; (ii) a specific derivation of the effective action for quadratic activation function, which makes no use of the Breuer-Major theorem; (iii) the generalization of the effective action in equation (4.7) to the case of multiple (but finite) outputs.

4.1.1 Link between Student's t-processes and shallow neural networks in the proportional limit

In obtaining the results reported in Sec. 4.1, our theory can be formulated as a statement on the probability distribution of the output variables

$$s^{\mu} \equiv \frac{1}{\sqrt{N_1}} \sum_{i_1=1}^{N_1} v_{i_1} \sigma(h_{i_1}^{\mu}) \,, \tag{4.19}$$

where $h \sim \mathcal{N}(0, C \otimes \mathbb{1}_{N_1})$, $v \sim \mathcal{N}(0, \lambda_1^{-1} \mathbb{1}_{N_1})$. Proceeding as in the derivation of the partition function presented in Methods, the p.d.f. of these variables can be written as a re-weighted Fourier transform,

$$P(s|\mathcal{T}_P) = \frac{e^{-\frac{\beta}{2}\sum_{\mu}(y^{\mu}-s^{\mu})^2}}{Z} \int \prod_{\mu} \frac{\mathrm{d}\bar{s}^{\mu}}{2\pi} \, e^{i\bar{s}^{\top}s} \Xi(\bar{s}) \,, \tag{4.20}$$

of the function

$$\Xi(\bar{s}) = \left(1 + \frac{1}{\lambda_1 N_1} \sum_{\mu,\nu}^{P} \bar{s}^{\mu} K_{\mu\nu}(C) \bar{s}^{\nu}\right)^{-\frac{N_1}{2}} . \tag{4.21}$$

It is straightforward to notice that as long as $N_1 \to \infty$ and $N_1 \gg P$, the dependence on N_1 disappears and we get:

$$\Xi(\bar{s}) \to e^{-\frac{1}{2\lambda_1}\sum_{\mu,\nu}^{P} \bar{s}^{\mu} K_{\mu\nu}(C) \bar{s}^{\nu}} . \tag{4.22}$$

This quantity has a very natural interpretation in view of the NNGP literature. Indeed, for N_1 large and P finite, the variables (4.19) are jointly multivariate Gaussian distributed according to the central limit theorem, as noted for example in [27]: this limit corresponds indeed to the RHS of our equation (4.22) and is the cornerstone of the mapping of an infinite-width Bayesian neural network to a GP. This is however no more the case when P is comparable to N_1: equation (4.21), derived exploiting the Gaussian equivalence based on the BM theorem in the proportional asymptotic limit $P/N_1 \sim O(1)$, is suggesting that the variables $\bar{s}^{\mu}$ are

distributed according to a multivariate Student's t-distribution [100–103].

The need of considering Student's t-processes as a generalization of NNGPs has been noted already in the case of different priors on the distribution of the last layer's weights [104]. Non-Gaussianity of the posterior in a form similar to that of Eq. (4.21) has appeared also in [85, 86, 105]. The reason why this kind of process arises in the case we are considering here can be understood with an heuristic argument: when N_1 and P are of the same order, we cannot take the limit $N_1 \to \infty$ before $P \to \infty$, and so we need to use the empirical covariance of the output variables s^μ instead of their true one in estimating their probability distribution. A more precise characterization of these neural network Student's t-processes (NNTPs) and the regime where they arise represent interesting topics for future work.

4.2 A tentative approach to study the replicated ensemble with shallow networks

The scaling regime define in Eq. 4.1, implies that the networks that we are considering work in the overparameterized regime, but this does not mean that the learning dynamics is convex. In fact, we know that the energy landscape of a neural network's weight configurations is in general non-convex [12, 47]. It is known and well discussed in the literature that this non-convexity, associated in some cases with hard phases of learning, can be explored with statistical mechanics tools in very simple architectures (perceptrons, commitee machines) using the escamotage of choosing binary weights. For such networks, biasing the dynamics to sample from the so-called *robust ensemble*, turns out to be beneficial both in terms of convergence time and in quality of solutions found [12, 47, 48, 55, 58]. In this section I present a first tentative approach to study the replicated ensemble with shallow 1HL networks in the infinite-width and proportional regime, with the ultimate goal of understanding if the robust ensemble remains relevant for learning in an overparameterized (but in general non-convex) learning task. This work only constitutes a preliminary analysis that still provide interesting results, summarized in the following. The detailed analytical derivations of all the results that

follow can be found in Appendix D.

Setting

Consider a system of Y interacting 1HL networks, parameterized by $\theta = (\theta_\alpha,\, \alpha = 1 \dots Y)$, each with function given by:

$$f_{\theta_\alpha}(x^\mu) = \frac{1}{\sqrt{N_1}} \sum_{i_1}^{N_1} v_{i_1}^\alpha \sigma \left(\frac{1}{\sqrt{N_0}} \sum_{i_0}^{N_0} w_{i_1 i_0}^\alpha x_{i_0}^\mu \right) \tag{4.23}$$

I want to describe a system in which the replicas interact through their weight vectors θ_α, though a quadratic interaction tuned by a parameter γ. The loss is defined as the sum of Y regularized MSE losses as defined in Eq. (4.2):

$$\begin{aligned} \mathcal{L}\left(\{\theta_\alpha\}\right) &= \frac{1}{2} \sum_{\alpha=1}^{Y} \sum_{\mu=1}^{P} \left(y^\mu - f_{\theta_\alpha}(x^\mu)\right)^2 + \mathcal{L}_{\text{REG}} \\ \mathcal{L}_{\text{REG}} &= \frac{\gamma}{4\beta} \sum_{\alpha\beta}^{Y} \sum_{i_0=1}^{N_0} \sum_{i_1}^{N_1} (w_{i_1 i_0}^\alpha - w_{i_1 i_0}^\beta)^2 + \frac{\lambda_0}{2\beta} \sum_{\alpha}^{Y} \sum_{i_0=1}^{N_0} \sum_{i_1}^{N_1} (w_{i_1 i_0}^\alpha)^2 + \\ &+ \frac{\lambda_1}{2\beta} \sum_{\alpha}^{Y} \sum_{i_1}^{N_1} (v_{i_1}^\alpha)^2 + \frac{\gamma}{4\beta} \sum_{\alpha,\beta}^{Y} \sum_{i_1=1}^{N_1} \left(v_{i_1}^\alpha - v_{i_1}^\beta\right)^2 . \end{aligned} \tag{4.24}$$

where the index μ runs over the examples in the dataset $\mathcal{D} = \{\mathbf{x}^\mu, y^\mu\}_{\mu=1\dots P}$One could consider the more general case of a quadratic interaction (in the replica index) between the weights:

$$\mathcal{L}_{\text{REG}} = \frac{\lambda_1}{2\beta} \sum_{\alpha,\beta}^{Y} \sum_{i_1=1}^{N_1} v_{i_1}^\alpha (M_1^{-1})_{\alpha\beta} v_{i_1}^\beta + \frac{\lambda_0}{2\beta} \sum_{\alpha\beta}^{Y} \sum_{i_0=1}^{N_0} \sum_{i_1}^{N_1} w_{i_1 i_0}^\alpha (M_0^{-1})_{\alpha\beta} w_{i_1 i_0}^\beta, \tag{4.25}$$

and re-obtain our desired regularized loss in Eq. (4.24) by imposing:

$$\begin{aligned} {M_0}^{-1} &= \frac{1}{\lambda_0} \left[\left(\lambda_0 + \gamma(Y-1)\right) \mathbb{I}_Y - \gamma \mathbb{O}_Y \right] \quad M_0 = \left(\frac{\gamma + \lambda_0}{\lambda_0 + Y\gamma} \mathbb{I}_Y + \frac{\gamma}{\lambda_0 + Y\gamma} \mathbb{O}_Y \right) \\ M_1^{-1} &= \frac{1}{\lambda_1} \left[\left(\lambda_1 + \gamma(Y-1)\right) \mathbb{I}_Y - \gamma \mathbb{O}_Y \right] \quad M_1 = \left(\frac{\gamma + \lambda_1}{\lambda_1 + Y\gamma} \mathbb{I}_Y + \frac{\gamma}{\lambda_1 + Y\gamma} \mathbb{O}_Y \right) \end{aligned} \tag{4.26}$$

where I have denoted $\mathbb{1}_Y$ the Y-dimensional identity matrix, and $\mathbb{0}_Y$ the Y-dimensional matrix with zeros on the diagonal and ones elsewhere.

Summary of the main results

It is possible to show (see Appendix D) that the computation of the partition function can be carried out, reducing it to a saddle point integral over (in principle) Y^2 order parameters.

$$
\begin{aligned}
Z &= \int \prod_{\alpha}^{Y} d\theta_\alpha e^{-\beta \mathcal{L}(\{\theta_\alpha\})} \propto \int \prod_{\alpha,\beta}^{Y} dQ_{\alpha\beta} e^{-\frac{N_1}{2} S(Q,\bar{Q})} \\
S(Q,\bar{Q}) &= -\mathrm{Tr}(Q\bar{Q}) + \log\det\left(\mathbb{1} + QM_1\right) + \\
&\quad + \frac{1}{N_1}\log\det\left(K^{(R)}(\bar{Q}) + \frac{\mathbb{1}_{P\times Y}}{\beta}\right) + \frac{1}{N_1} y^T \sum_{\alpha\beta}\left(K^{(R)}(\bar{Q}) + \frac{\mathbb{1}_{P\times Y}}{\beta}\right)^{-1}_{\alpha\beta} y
\end{aligned}
\tag{4.27}
$$

with the renormalized kernel $K^{(\mathrm{R})}(\bar{Q})$ given by:

$$
\left[K^{(\mathrm{R})}(\bar{Q})\right]^{\alpha\beta}_{\mu\nu} = \frac{\bar{Q}_{\alpha\beta}}{\lambda_1} K^{\mathrm{REP}}_{\alpha\beta\mu\nu} \tag{4.28}
$$

The dynamics is governed by a kernel K^{REP}, that inherits a replica-symmetric (RS) structure from the replicas themselves (that we consider indistinguishable).

$$
K^{\mathrm{REP}} = K^1 \otimes \mathbb{1}_Y + K^0 \otimes \mathbb{0}_Y \tag{4.29}
$$

$$
K^1_{\mu\nu} = \int D\mathbf{h}\, \sigma(h^{\mu\alpha})\sigma(h^{\nu\alpha}) \tag{4.30}
$$

$$
K^0_{\mu\nu} = \int D\mathbf{h}\, \sigma(h^{\mu\alpha})\sigma(h^{\nu\beta}) \tag{4.31}
$$

where $D\mathbf{h} \equiv p(\{h_{\mu\alpha}\}) \prod_{\mu\alpha} \mathrm{d}h_{\mu\alpha}$. Consistently with [95], we find that the preactivations of the networks after training are gaussian distributed:

$$
p(\{\mathbf{h}\}) = \mathcal{N}(0,\Gamma) \qquad \Gamma = C \otimes M_0 \tag{4.32}
$$

where $\otimes$ denotes the outer matrix product, and M_0 defined in (4.26).

Predictor statistic and accuracy in the infinite-width limit

In the infinite width limit where $N_1 \gg P$, only the first two terms count in the replicated partition function (4.27). The saddle point solution is easily found as:

$$Q^*_{\alpha\beta} = 0 \quad \forall \alpha\beta, \qquad \bar{Q}^* = M_1 \tag{4.33}$$

Note that this is completely different from the unreplicated case, where we have that $M_1 = M_0 = \mathbb{1}$, and there is no contribution from the off-diagonal terms of $\bar{Q}$, and therefore from K^0. In this framework, the statistics of the output after training is a multivariate Gaussian distribution in the replica space:

$$p\left(\{f_{\theta_\alpha}\})|\mathcal{D}\right) = \mathcal{N}\left(\mathbf{m}, \Sigma\right) \tag{4.34}$$

with $\mathbf{m} = (m_\alpha, \alpha = 1 \dots Y)$, and Σ a $Y \times Y$ matrix:

$$m^\alpha = \sum_{\alpha'\mu\nu} y^\mu [\kappa^{(R)}]^{\alpha'\alpha}_\nu \left[\left(K^{(\mathrm{R})} + \frac{\mathbb{1}}{\beta}\right)^{-1}\right]^{\alpha'\alpha}_{\mu\nu} \tag{4.35}$$

$$\Sigma^{\alpha\beta} = [\kappa^{(R)}]^{\alpha\beta}_0 - \sum_{\mu\nu} [\kappa^{(R)}]^{\alpha\beta}_\mu \left[\left(K^{(\mathrm{R})} + \frac{\mathbb{1}}{\beta}\right)^{-1}\right]^{\alpha\beta}_{\mu\nu} [\kappa^{(R)}]^{\alpha\beta}_\nu \tag{4.36}$$

where the renormalized kernel is given in Eq, (4.28) and, analogously to the unreplicated case, the Kernel function must be evaluated at the unseen example $\mathbf{x}^0$, yielding the following extra terms: with:

$$[\kappa^{(R)}]^{\alpha\beta}_\mu = \frac{\bar{Q}_{\alpha\beta}}{\lambda_1} \kappa^{\mathrm{REP}}_{\alpha\beta\mu}, \qquad \kappa^{\mathrm{REP}} = \kappa^1 \otimes \mathbb{1}_Y + \kappa^0 \otimes \mathbb{0}_Y \tag{4.37}$$

$$\kappa^1_{\alpha\beta\mu} = \int D\mathbf{h}\, \sigma(h^{\mu\alpha})\sigma(h^{\nu\alpha}) \tag{4.38}$$

$$\kappa^0_{\alpha\beta\mu\nu} = \int D\mathbf{h}\, \sigma(h^{\mu\alpha})\sigma(h^{\nu\beta}) \tag{4.39}$$

Instead of the test loss, here I choose to compute another observable to test the network's performances after training, that is the test (generalization) accuracy ϵ_{acc}. For an ensemble of

Y interacting replicas, this reads:

$$\epsilon_{acc} = \left\langle \theta \left(\frac{1}{Y} \sum_{\alpha} y^0 f_{\theta_\alpha}(x^0) \right) \right\rangle , \tag{4.40}$$

where the average is performed over the Gibbs ensemble associated to the replicated loss (4.24) at zero temperature, and θ is the Heaviside step function. This metric, suitable for classificiation problems, gives the expected percentage of trained netorks that will correctly label unseen example $\mathbf{x}^0, y^0$.

$$\epsilon_{acc} = \left\langle \theta \left(\frac{1}{Y} \sum_{\alpha} y^0 f_{\theta_\alpha}(x^0) \right) \right\rangle = H \left(\frac{-\text{sign}(y^0) \sum_\alpha m^\alpha}{\sqrt{\sum_{\alpha\beta} \Sigma_{\alpha\beta}}} \right) ,$$

and the $H(\cdot)$ function is defined as $H(x) = \frac{1}{2}\text{Erfc}\left(\frac{x}{\sqrt{2}}\right)$.The covariance and mean of the output distribution read:

Note that for $\gamma = 0$, that is the case of non-interacting replicas, the accuracy reads:

$$\epsilon_{acc} = H \left(\frac{-\text{sign}(y^0)\Gamma}{\sqrt{\sigma^2}/Y} \right) \Big|_{Q^* \bar{Q}^*} \tag{4.41}$$

with Γ and σ^2 consistent with the ones found in [95], with the number of trials Y rescaling the variance.

4.3 Asymptotic effective action for Bayesian deep fully-connected architectures

In the generic case of a deep fully-connected architecture with a finite number of layers L and zero-mean activation function, the partition function can be expressed in terms of a $2L$-dimensional integral. In this section I will Detailes of this calculations can be found in below

in section B.2.3 :

$$Z_{\text{DNN}} = \int \prod_{\ell=1}^{L} \mathrm{d}Q_\ell \, \mathrm{d}\bar{Q}_\ell e^{-\frac{N_L}{2} S_{\text{DNN}}(\{Q_\ell, \bar{Q}_\ell\})} \,, \tag{4.42}$$

where the effective action is given by:

$$\begin{aligned} S_{\text{DNN}} = &\sum_{\ell=1}^{L} \frac{\alpha_L}{\alpha_\ell} \left[-Q_\ell \bar{Q}_\ell + \log(1 + Q_\ell) \right] \\ &+ \frac{\alpha_L}{P} \text{Tr} \log \beta \left(\frac{\mathbb{1}}{\beta} + K_L^{(R)}(\{\bar{Q}_\ell\}) \right) + \frac{\alpha_L}{P} y^T \left(\frac{\mathbb{1}}{\beta} + K_L^{(R)}(\{\bar{Q}_\ell\}) \right)^{-1} y \end{aligned} \tag{4.43}$$

and we have introduced a renormalized kernel $K^{(R)}$ that generalizes the the recurrence relation for the L-layer NNGP kernel as:

$$K_\ell^{(R)}(\{\bar{Q}_\ell\}) = \bar{Q}_\ell / \lambda_\ell K \circ \left[K_{\ell-1}^{(R)}(\{\bar{Q}_\ell\}) \right] \,, \quad K_0^{(R)} = C \,, \tag{4.44}$$

where C is the covariance matrix of the inputs defined in Eq. (1.40). It is important to stress that each $K_\ell^{(R)}$ depends on the variables $\bar{Q}_1, \ldots, \bar{Q}_{\ell-1}$ only. For completeness, we notice that the recurrence relation for the infinite-width kernel K_L is given by equation (4.44) with $\bar{Q}_\ell = 1$ $\forall \ell = 1, \ldots, L$.

This action shares the same structure as the one found in section 4.1 for the special case of 1HL, with the difference that for L hidden layers, the recursive nature of the derivation introduces additional order parameters that are nested in the definition of the kernel K_L. Furthermore, since our derivation applies to layers of arbitrary size N_ℓ, the action also depends on the aspect ratios α_ℓ. The computation of the generalization error over a new example $(\mathbf{x}^0, y^0)$ gives:

$$\langle \epsilon_{\text{g}}(\mathbf{x}^0, y^0) \rangle = (y^0 - \Gamma_L)^2 + \sigma_L^2 \tag{4.45}$$

$$\Gamma_L = \sum_{\mu\nu} \kappa_{L\mu}^{(R)} \left(\frac{\mathbb{1}}{\beta} + K_L^{(R)}(\{\bar{Q}_\ell\}) \right)^{-1}_{\mu\nu} y_\nu, \tag{4.46}$$

$$\sigma_L^2 = \kappa_{L0}^{(R)} - \sum_{\mu\nu} \kappa_{L\mu}^{(R)} \left(\frac{\mathbb{1}}{\beta} + K_L^{(R)}(\{\bar{Q}_\ell\}) \right)^{-1}_{\mu\nu} \kappa_{L\nu}^{(R)} \tag{4.47}$$

where $\kappa^{(R)}_{L\mu}$, $\kappa^{(R)}_{L0}$ are recursive kernels computed from the recurrence given in equation (4.44) using the input $\mathbf{x}^0$ in the initial conditions.

Additional results

In the Appendix C I show how to derive a series of additional results: (i) a generalization of this effective action for finite-mean activation functions; (ii) a way to recover the linear case in the isotropic limit $\alpha_\ell = \alpha \; \forall \ell = 1, \dots, L$; (iii) how to use (i) to correct the heuristic theory for ReLU activation presented in Ref. [9].

4.4 Shallow Convolutional and Locally connected networks

In this section I will present the resulting effective action in the proportional regime for locally connected shallow architectures, with and without weight sharing. The details of the analytical derivations can be found in the Appendix E. A 1HL NN with one-dimensional convolutions implements the following function:

$$
\begin{aligned}
f_{\text{CNN}}(\mathbf{x}) &= \frac{1}{\sqrt{\lambda_1 N_1}} \sum_{i=1}^{\lfloor N_0/S \rfloor} \sum_{a=1}^{N_c} v_i^a \sigma\left[h_i^a(\mathbf{x})\right] \,, \\
h_i^a(\mathbf{x}) &= \frac{1}{\sqrt{\lambda_0 M}} \sum_{m=-\lfloor M/2 \rfloor}^{\lfloor M/2 \rfloor} W_m^a x_{Si+m} \,.
\end{aligned} \tag{4.48}
$$

Here M is the dimension of the channel mask, S is the stride, the index $i = 1, \dots, \lfloor N_0/S \rfloor$ runs over the input coordinates, the index $a = 1, \dots, N_c$ runs over the channels and the index m moves through the spatial mask of the convolutional filter. where σ is a odd activation function and the v's are the learnable weights of the readout layer. The locally connected network LCN, has only one of the properties of a CNN, which is locality. The absence of weight sharing means that the convolutional mask is not unique and parsed on the input, but

there are as many masks as $\lfloor N_0/S \rfloor$. For LCNs, we have:

$$f_{\text{LCN}}(x) = \frac{1}{\sqrt{N_c \lfloor N_0/S \rfloor}} \sum_{i=1}^{\lfloor N_0/S \rfloor} \sum_{a=1}^{N_c} v_i^a \sigma\left(h_i^a\right) \tag{4.49}$$

$$h_i^a(x) = \sum_{m=-\lfloor M/2 \rfloor}^{\lfloor M/2 \rfloor} \frac{W_{im}^a x_{Si+m}}{\sqrt{M}} \tag{4.50}$$

Our goal is to derive an effective action in the same setting of Refs. [9, 95], in the thermodynamic limit described in (4.1), where $(N_c, P) \to \infty$ and their ratio $\alpha_c = P/N_c$ is finite.

4.4.1 Finite-width effective action: how the feature matrix Q_{ij} is exploited in convolutional and LC networks.

The partition function for the learning problem with the 1Hl CNN can be approximated as an integral over an ensemble of $\lfloor N_0/S \rfloor \times \lfloor N_0/S \rfloor$ matrices Q and $\bar{Q}$

$$Z_{\text{CNN}} = \int \mathcal{D}Q\mathcal{D}\bar{Q} e^{-N_c/2 S_{\text{CNN}}(Q,\bar{Q})}, \tag{4.51}$$

where the effective action is given by:

$$S_{\text{CNN}}(Q,\bar{Q}) \equiv -\sum_{ij} Q_{ij}\bar{Q}_{ij} + \text{Tr}\log(\mathbb{1}_{\lfloor \frac{N_0}{S} \rfloor} + Q) + \tag{4.52}$$

$$+ \frac{\alpha_c}{P}\text{Tr}\log\beta\Big(\frac{\mathbb{1}_P}{\beta} + K_{\text{CNN}}^{(\text{R})}\Big) + \frac{\alpha_c}{P} y^\top \Big(\frac{\mathbb{1}_P}{\beta} + K_{\text{CNN}}^{(\text{R})}\Big)^{-1} y. \tag{4.53}$$

The trace in the first two terms is over $\lfloor N_0/S \rfloor \times \lfloor N_0/S \rfloor$ operators, whereas the renormalized kernel $K_{\text{CNN}}^{(\text{R})}$ is a $P \times P$ matrix and therefore the trace in the third term and the scalar products with the output labels lie in a P-dimensional vectorial space. The matrix elements of the renormalized kernel are given by:

$$\Big[K_{\text{CNN}}^{(\text{R})}(\bar{Q})\Big]_{\mu\nu} = \frac{1}{\lambda_1 \lfloor N_0/S \rfloor} \sum_{ij=1}^{\lfloor N_0/S \rfloor} \bar{Q}_{ij} K_{\mu\nu}^{ij}\,, \tag{4.54}$$

where the kernel $K^{ij}_{\mu\nu}$ is given in terms of the elements of the following local covariance matrix

$$C^{ij}_{\mu\nu} = \frac{1}{\lambda_0 M} \sum_{m=-\lfloor M/2 \rfloor}^{\lfloor M/2 \rfloor} x^{\mu}_{Si+m} x^{\nu}_{Sj+m}, \tag{4.55}$$

with λ_0 being the hidden layer Gaussian prior, via the same functional relations of the FC case:

$$K^{ij}_{\mu\nu} = \int d^2t \, \mathcal{N}_t \left(0, \tilde{C}^{ij}_{\mu\nu}\right) \sigma(t_1)\sigma(t_2)\,, \qquad \tilde{C}^{ij}_{\mu\nu} = \begin{pmatrix} C^{ii}_{\mu\mu} & C^{ij}_{\mu\nu} \\ C^{ij}_{\mu\nu} & C^{jj}_{\nu\nu}\,. \end{pmatrix}.$$

It is worth noticing that this is the same local kernel that has been already found in the seminal work on the GP limit of infinite-width CNNs [29], as discussed in Chapter 2. In our effective action framework we recover that result by solving the saddle-point equations for the matrix $\bar{Q}$ in the limit of $\alpha_c \to 0$, where one finds $\bar{Q}_{ij} = \delta_{ij}$. Note that only the diagonal components of the local kernel contribute to the prediction in this limit, and there are no free parameters that the network can optimize.

The case of locally connected networks is somewhere in between: it is still true that only the diagonal components Q_i of Q contribute to the prediction, but they are free to be optimized. More precisely, the partition function for a LCN shares the same structure as the FC and CNN ones:

$$Z = \int \prod_i dQ_i d\bar{Q}_i e^{-\frac{N_c}{2} S_{\text{LCN}}(Q,\bar{(Q)}}, \tag{4.56}$$

where the LCN effective action S_{LCN} is given by:

$$S_{\text{LCN}} \equiv -\sum_i (Q_i\bar{Q}_i - \log(1+Q_i)) + \tag{4.57}$$

$$+ \frac{\alpha_c}{P} \text{Tr} \log \beta \Big(\frac{\mathbb{1}_P}{\beta} + K^{(\text{R})}_{\text{LCN}} \Big) + \frac{\alpha_c}{P} y^{\top} \Big(\frac{\mathbb{1}_P}{\beta} + K^{(\text{R})}_{\text{LCN}} \Big)^{-1} y. \tag{4.58}$$

with the LCN renormalized kernel:

$$\left[K^{(\mathrm{R})}_{\mathrm{LCN}}(\bar{Q})\right]_{\mu\nu} \equiv \frac{1}{\lfloor N_0/S \rfloor} \sum_{i=1}^{\lfloor N_0/S \rfloor} \bar{Q}_i K^i_{\mu\nu} \,. \tag{4.59}$$

Note that this action shares the same functional form as the one for FCN and CNN, but the renormalized Kernel contains only the diagonal elements of the local kernel matrix defined in Eq. (4.54). A detailed description of how this result is obtained can be found in Appendix D.

CHAPTER 5

Physical implications

In the final Chapter of this work, I will discuss some physical consequences of the statistical mechanics framework that I have previously introduced. I will firstly test a number of predictions that stem from the effective actions derived in Chapter 4, from the generalization performance to more general considerations on the monotonicity of the learning curves of different architectures [95, 106]. Finally, I will illustrate a mechanism for feature learning, a consequence of this statistical mechanics description, that can take place in CNNs but not in FCNs or in LCNs without weight sharing [106]. To conclude, I will provide numerical evidence for this mechanism, called *local kernel renormalization*.

5.1 Predictive power of the theory

5.1.1 Zero-temperature generalization performance of 1HL networks

The most straightforward prediction of the previously discussed framework is the zero-temperature test error for 1HL networks. Computed as an observable over the Gibbs ensemble

Architecture	Kernel type	IW Kernel	Renormalized Kernel
FC	$K_{\mu\nu}$	$K_{\mu\nu}$	$\bar{Q}K_{\mu\nu}$
LCN	$K^{ii}_{\mu\nu}$	$\sum_i K^{ii}_{\mu\nu}$	$\sum_i \bar{Q}_i K^{ii}_{\mu\nu}$
CNN	$K^{ij}_{\mu\nu}$	$\sum_i K^{ii}_{\mu\nu}$	$\sum_{ij} \bar{Q}_{ij} K^{ij}_{\mu\nu}$

Table 5.1 –Summary of the analytical results for one hidden layer networks. We identify three different types of kernels in the Bayesian effective action for FC, LC and convolutional networks: a global kernel in the FC case, a local diagonal kernel in LCNs and a local (with additional non-diagonal components) in CNNs. The infinite-width kernel that describes LCNs and CNNs is the same, as shown in [29]. In the proportional limit, the renormalization of the FC kernel is scalar, i.e. the NNGP kernel is only globally rescaled by a data-dependent number $\bar{Q}$. In LCNs, we can adjust a vector of components $\bar{Q}_i$, which determines how much the self-correlations of the i-th patch of the diagonal local kernel do contribute to the final prediction. In CNNs, due to locality and weight-sharing, we can fine-tune a full matrix $\bar{Q}_{ij}$, which controls how much the local correlations of patches i and j will enter the effective kernel learned by the network after training. This is what we call *local kernel renormalization.*

(1.32), we have seen it can be expressed in terms of the usual bias-variance decomposition:

$$\begin{aligned}
\langle \epsilon_{\rm g}(\mathbf{x}^0, y^0)\rangle &= \left(y^0 - \Gamma_1\right)^2 + \sigma_1^2\,, \\
\Gamma_1 &= \sum_{\mu,\nu} \kappa_\mu(\mathbf{x}^0) K^{-1}_{\mu\nu}\; y_\nu\,, \\
\sigma_1^2 &= \frac{\bar{Q}^*}{\lambda_1}\left[\kappa_0(\mathbf{x}^0) - \sum_{\mu,\nu} \kappa_\mu(\mathbf{x}^0) K^{-1}_{\mu\nu}\; \kappa_\nu(\mathbf{x}^0)\right]\,,
\end{aligned} \tag{5.1}$$

where the NNGP kernel K is defined in (1.43), and $\kappa_\mu(\mathbf{x}^0) \equiv K(\mathbf{x}^\mu, \mathbf{x}^0)$, $\kappa_0(\mathbf{x}^0) \equiv K(\mathbf{x}^0, \mathbf{x}^0)$ can be computed from the functional definition of the NNGP kernel in Eq. (1.42) using the new unseen input $\mathbf{x}^0$.

One can directly employ equation (5.1) to obtain testable predictions for the generalization error of finite-width 1HL architectures trained in the Bayesian learning setting; results of this comparison are shown in panel (a) of Fig. 5.1 for two specific regression tasks defined on the CIFAR10 and MNIST datasets (details on the numerical experiments are provided in the Appendix Section F.1). It turns out that the generalization curves for the two regression tasks are monotonically increasing (decreasing) as a function of N_1 depending on the fact that the observable $y^\top \left(K/\lambda_1\right)^{-1} y/P$ is smaller (larger) than one. The importance of this quantity in

controlling the generalization performance has been already noted in linear networks [9, 80] as well as in direct perturbation theory at finite P for non-linear networks [107, 108].

Two semi-quantitative predictions for the general behavior of the generalization error can also be easily tested, just by looking at the dependence of equation (5.1) on the size of the hidden layer N_1 and on the Gaussian prior of the last layer λ_1. At $T = 0$, the bias is constant as a function of N_1 (as explicitly observed also in the linear case in Ref. [9]) and of λ_1. On the contrary, the variance depends on N_1 and decreases as $1/\sqrt{\lambda_1}$ in the large-λ_1 limit. These observations lead to the following two testable predictions: (i) increasing the magnitude of the Gaussian prior λ_1 should systematically improve the generalization performance at any N_1; (ii) for large λ_1 the dependence on N_1 of the generalization error should disappear (see also the numerical experiments performed in panel (b) in Fig. 5.1).

5.1.2 Finite-temperature generalization loss in 1HL networks

The more general expression of the generalization loss at generic temperature, given in Eq. (4.18), can be evaluated by numerically minimising the action (4.7) at given temperature $T = \beta$. In Figure 5.2, I report the result of experiments with finite neural networks trained at finite temperature, in comparison with the theory prediction computed from (4.18). The accordance between theory and experiments is excellent in all the settings that we explored, highlighting the non-trivial effects of finite temperature T on the generalization performances. For instance, depending on the dataset into consideration (CIFAR or MNIST), it can or cannot be beneficial - in terms of test loss - to work at finite temperature. This discussion is part of preliminary work for a project I am currently doing in collaboration with the Statistical Physics group in Parma, and will be the object of discussion in future work.

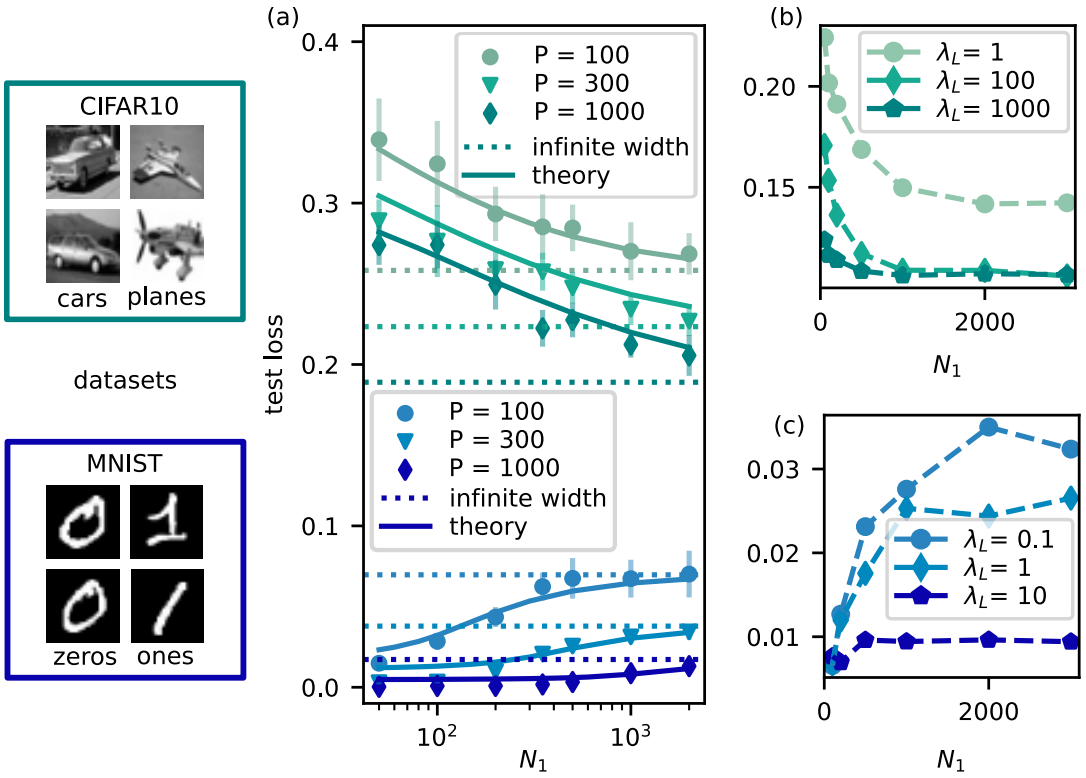


Figure 5.1 –(a) Learning curves of 1HL architectures with Erf activation (trained with a discretized Langevin dynamics, see Appendix F) as a function of the hidden layer size N_1 for two regression tasks on the CIFAR10 (above) and MNIST (below) datasets with zero/one labels. The experimental test loss at different values of the trainset size P (points with error bars indicating one standard deviation) are compared with the theory computed from equation (4.18) (solid lines). (b,c) Experimental learning curves as a function of N_1 for increasing values of the Gaussian prior of the last layer λ_1. Dashed lines connecting the points are shown to guide the eye. The nets are trained on $P = 3000$ examples from the CIFAR10 dataset in (b) and $P = 500$ examples from MNIST in (c). Two qualitative predictions of the theory at zero temperature are checked: (i) the generalization loss should decrease for any N_1 when λ_1 grows; (ii) the dependence of the learning curves on N_1 disappears in the large-λ_1 limit, since the bias is constant (see also main text).

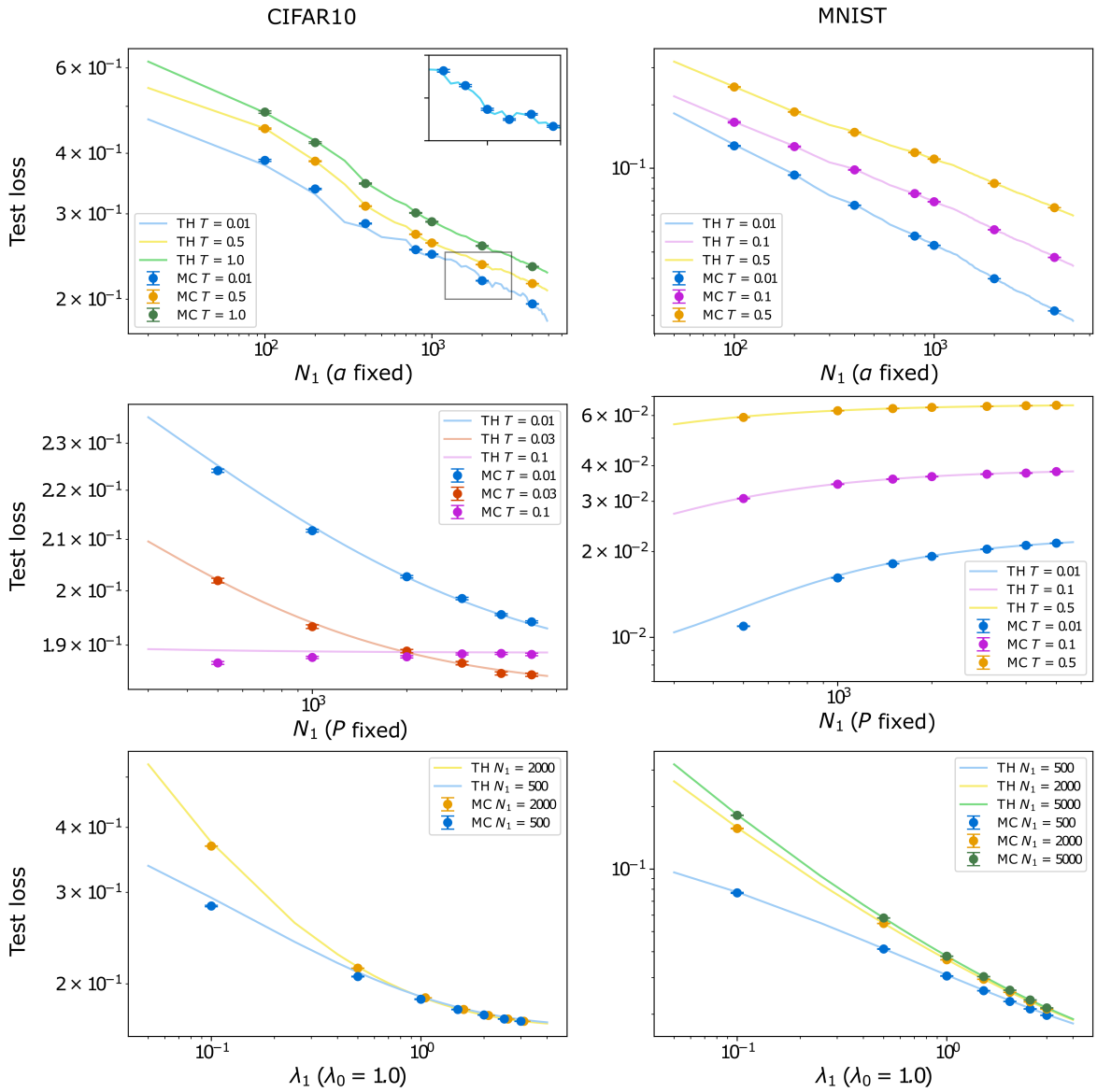


Figure 5.2 –(a) Learning curves of 1HL architectures with Erf activation, trained with a discretized Langevin dynamics on two regression tasks on the CIFAR10 (left) and MNIST (right) datasets (points with bars), compared to the predicted theory derived from equation 4.18 for different temperatures $T = 1/\beta$ (solid lines).. (above) The experimental test loss is plotted as a function of the fixed ratio $\alpha = P/N_1$. Keeping α fixed implies that the trainset has to be slightly changed at each simulation. This feature is visible in the ruggedness of theoretical curves. The inset shows that our theory is able to capture the small nuances brought by different training sets to the generalization error, perfectly accomodating the result of experiments for each dataset. (middle) Experimental test loss is computed at fixed P, with varying values of N_1. Raising the temperature has a detrimental effect for the MNIST dataset (right), but seems to be beneficial in a certain range for the CIFAR dataset. (below) The agreement between theoretical test loss and experimental one remains spectacular also for different values of the last-layer Gaussian prior λ_1 (keeping λ_0 fixed).

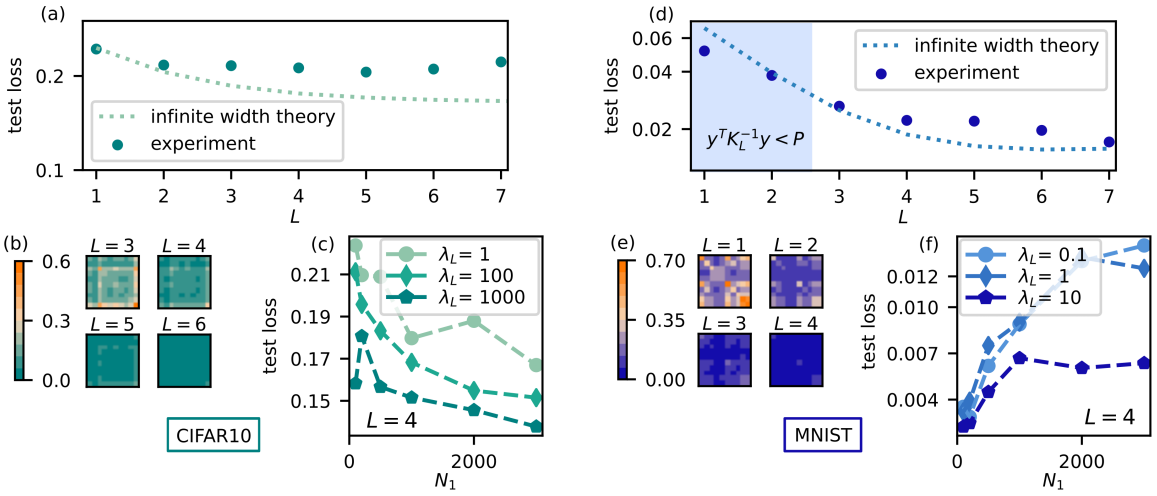


Figure 5.3 –(a,d) Test loss of a L-HL neural network with ReLU activation, as a function of the depth L, for $P = 100$. The net is trained on a regression task in the small α regime ($\alpha = 0.1$), close to the infinite-width limit. The finite-width network can outperform the infinite-width prediction only when $s_L < 1$ (shaded area), i.e. only for the MNIST task and for depth $L < 3$. (b,e) Visualization of the entries of the infinite-width NNGP kernel at different layers of the network. The ReLU NNGP kernel converges to zero after repeated iterations. This generates almost vanishing eigenvalues that makes s_L eventually always larger than one. (c,f) Test loss of a 4-HL network trained on $P = 1000$ examples with different regularization strengths (with $N_\ell = N = 1000$). While increasing the magnitude of the Gaussian prior of the last layer still improves generalization for all N, it is not clear anymore (as it was for 1HL networks) that the curve at large λ_L is a constant as a function of N. The dashed line is shown to guide the eye.

5.1.3 Prediction for L-layer networks

Inspection of the generalization error of deep FC networks given in Equations (4.45) and (4.47), allows to perform a scaling analysis of the dependence of the generalization error on the Gaussian prior in the last layer λ_L (in the zero temperature limit). It turns out that the bias does not depend on it, whereas the variance σ_L^2 approaches zero as $1/\sqrt{\lambda_L}$ as λ_L is taken to infinity. This means that also in the case of finite depth $L > 1$, training at large values of the Gaussian prior of the last layer should improve generalization at any aspect ratio of the network. This general observation is confirmed by numerical experiments in panels (c) and (f) of Fig. 5.3. However, differently from the 1HL case, the bias does depend on the aspect ratio even in the zero-temperature limit and we cannot expect anymore that the dependence on the aspect ratios of the networks α_ℓ disappears in the $\lambda_L \to \infty$ limit. One can obtain another prediction of the theory at L layers (that again confirms previous results

on linear networks and perturbative calculations for non-linear networks [9, 80, 107, 108]) by considering the effective action for ReLU activation. A straightforward Taylor expansion around the infinite-width limit $\alpha_\ell = \alpha = 0 \; \forall \ell = 1, \dots, L$ shows that the first correction to the test loss $\Delta\epsilon_{\text{g}}$ is proportional to:

$$\Delta\epsilon_{\text{g}} \propto \alpha \left(\frac{1}{P} y^T K_L^{-1} y - 1 \right) . \tag{5.2}$$

where K_L is the solution of recurrence in equation (4.44) for $\bar{Q}_\ell = 1 \; \forall \ell = 1, \dots, L$ and ReLU activation. This means that there exists a simple scalar observable that determines whether the finite-width deep neural network will outperform its infinite-width counterpart that generalises the one found at 1HL:

$$s_L = \frac{1}{P} y^T K_L^{-1} y \, . \tag{5.3}$$

In particular, the finite-width network is expected to outperform its infinite-width counterpart whenever $s_L < 1$. In panel (a) and (c) of Fig. 5.3 this prediction is checked for deep architectures with ReLU activation on the same regression tasks employed in the 1HL case. Notice that s_L quickly diverges to infinity as the number of hidden layers L grows. The reason for this is simply that the ReLU NNGP kernel K_L develops at least one zero eigenvalue as $L \to \infty$. This ultimately occurs because each element of the matrix K_L converges to zero as L grows (see panel (b) and (c) of Fig. 5.3), as one can easily check by looking at the explicit recurrence relation for the NNGP ReLU kernel. [27, 109]. Note that this singularity can be equivalently thought as the fixed point of the discrete dynamical map defined by the recurrence relation for the NNGP kernel and therefore it might be worth investigating the relation between the generalization performance in our asymptotic limit and the line of work on the edge of chaos in random neural networks [110, 111].

Equation (5.2) provides an additional link with Student's t inference. In fact, the same criterion has been found by Tracey and Wolpert [112] in the study of Bayesian optimization with Student's t-processes. Here the authors show that the value of s_L determines whether the

Student's t-process they consider has a larger/smaller variance than the corresponding GP with the same kernel.

5.2 Finite-width 1HL FCNs cannot outperform infinite-width GP kernels in the proportional regime

A straightforward consequence of the theoretical framework presented in the previous Chapters is that a shallow (1HL) FC network will always underperform with respect to a fine-tuned 1HL wide network. The only parameters that one can freely chose in such architectures are the Gaussian priors of respectively the first and second hidden layers (λ_0, λ_1), and the fine-tuning therefore depends on those quantities. A series of natural observations lead to this result:

1. The renormalized kernel $K^{(\mathrm{R})}$ enters in the predictor's statistics for a new unseen test element exactly in the same way the NNGP kernel does in the infinite-width limit, and thus it completely determines the generalization performance at finite width, once the saddle-point equations for the parameter $\bar{Q}$ are solved [95].

2. The scalar parameter $\bar{Q}$ always appears in combination with the corresponding Gaussian prior λ_1 as $\bar{Q}/\lambda_1$. As such this means that the parameter $\bar{Q}$, once evaluated on the saddle-point, is just data-dependent scalar renormalization of the Gaussian prior.

This implies that, once the size of the training set and the activation function σ are fixed, it is always possible to re-obtain the performance of any finite-width network just by carefully fine-tuning of the Gaussian prior λ_1 in the corresponding infinite-width kernel. Therefore, the generalization performance of any finite-width 1HL FC network is bounded by the one of a suitable infinite-width kernel with optimal choice of the Gaussian prior.

The empirical observation that infinite-width 1HL FCNs seem to systematically outperform their finite-width counterpart –which now finds a possible explanation in this framework– points at the somewhat disappointing consequence that feature learning is not particularly effective in networks with FC layers alone at finite width.

Let me stress that the aforementioned observation is not ruling out at all other possible forms of feature learning in FC architectures: (i) Renormalization of the infinite-width kernel is not the only source of feature learning possible. Higher order kernels, irrelevant in the infinite-width limit, may play a role in feature learning, especially as long as one considers the case where the size of the dataset P roughly scales as the number of parameters $\sim L \times N^2$ ($N_\ell = N \; \forall \ell$) of the FC DNN. The recent work [14] is a notable example of how a FC network can learn a convolutional structure in special settings; (ii) Our result holds for Bayesian learning, i.e. when the weights of the networks are sampled from the canonical Gibbs ensemble. This is only obtained if training is governed by a Markov chain Monte Carlo. Practical learning algorithms with state-of-the-art optimizers may behave differently and possibly activate alternative forms of feature learning; (iii) here we are limiting our analysis to the standard parameterization setting where the last layer is normalized as $1/\sqrt{N_1}$ and we cannot say that much about the mechanism for feature learning that may occur in the mean field regime [64] where the last layer is normalized as $1/N_1$ (we note that the framework of [11] seems more suitable to deal with this case). Nonetheless, it must be also said that the empirical evidence provided in Refs. [27, 29, 72, 89, 94] is quite against the fact that these forms of feature learning can play a significant role in improving the generalization performance of FC networks.

Let us now turn our focus to the second question raised in Chapter 3 of this manuscript. How –contrarily to what occurs in deep nets with FC layers only– do deep architectures with convolutional layers exploit the finite-width regime for efficient feature learning? This fact can be rationalized in view of the previous discussion on CNNs. The form of the renormalized kernel obtained in (5.15) suggests that CNNs at finite width could be able, in some way, to break the global (trivial) renormalization of the infinite-width kernel that occurs in FC deep networks.

5.3 A simple mechanism for feature learning in finite-width CNNs: Local Kernel renormalization

Let me now analyze the saddle-point equations deriving from Eq. (4.53). Analytical insight can be gained in the zero temperature limit, where one finds the following matrix equations that determine Q and $\bar{Q}$:

$$\bar{Q}_{ij} = (\mathbb{1} + Q)^{-1}_{ij} \tag{5.4}$$

$$Q_{ij} = \frac{\alpha_c}{P}\mathrm{Tr}\Big(K^{ij}\left[K^{(\mathrm{R})}_{\mathrm{CNN}}(\bar{Q})\right]^{-1}\Big) + \\ - \frac{\alpha_c}{P} y^T\Big(\left[K^{(\mathrm{R})}_{\mathrm{CNN}}(\bar{Q})\right]^{-1} K^{ij}\left[K^{(\mathrm{R})}_{\mathrm{CNN}}(\bar{Q})\right]^{-1}\Big) y\,. \tag{5.5}$$

Here the shortcut K^{ij} has been introduced to indicate the partial trace over indices (μ, ν) in the local kernel defined in Eq. (4.54). More explicitly, given a matrix O with elements $O_{\mu\nu}$, the partial trace is defined as $\mathrm{Tr}(K^{ij}O) := \sum_{\mu\nu=1}^{P} K^{ij}_{\mu\nu}O_{\mu\nu}$. The exact solution of these matrix equations cannot be obtained in closed form. However it is possible to compute it perturbatively around the infinite-width limit $\alpha_c \to 0$. Parametrizing the solution of the first equation as $\bar{Q}_{ij} = \delta_{ij} + \alpha_c \delta\bar{Q}_{ij} + o(\alpha_c)$ yields the simple result:

$$\delta\bar{Q}_{ij} = \frac{1}{P}\tilde{y}^T K^{ij}\tilde{y} - \frac{1}{P}\mathrm{Tr}\left(K^{ij}\bar{K}^{-1}\right)\,, \tag{5.6}$$

where $\tilde{y}^\mu = \sum_{\mu=1}^{P} \bar{K}^{-1}_{\mu\nu} y^\mu$ and $\bar{K}$ is the infinite-width GP-CNN averaged kernel defined in Eq. (2.28).

Providing a physical interpretation of this finding is one of the main results of the current work. Our claim is that the matrix $\bar{Q}$ provides a compact description of feature learning in the CNN model under consideration. In this specific architecture the indices of the feature matrix $\bar{Q}$ are in one-to-one correspondence with the $\lfloor N_0/S \rfloor$ patches of the local covariance matrix defined from the trainset elements x^μ. From the first term of the r.h.s. of Eq. (5.6) we can notice that whenever the local kernel of the CNN at the spatial locations (i, j) will have a significant overlap with the effective label vector $\tilde{y}$, the element $\bar{Q}_{ij}$ of the feature matrix $\bar{Q}$

will differ from one. In other words, this is a measure of the relative importance of the pairwise spatial correlations in the input dataset wrt the labels. From Eq. (4.54), we can now interpret the matrix $\bar{Q}$ as a feature, data-dependent matrix that tells us how much a given component of the local kernel contribute to the renormalized kernel $K^{(\mathrm{R})}_{\mathrm{CNN}}$.

The local kernel renormalization that takes place in this CNN toy model in the proportional limit is completely different from the trivial global one that occurs in the corresponding FC 1HL network. At this point we can ask ourselves whether either the ingredients in CNNs design, local connectivity and weight sharing, are needed to observe the phenomenon of local kernel renormalization (LKR). The answer is affirmative: LKR does not occur in a shallow network with local connectivity only, as we explicitly check in the supplemental material for the same toy model employed here without weight sharing. In that case, only the diagonal components of the local kernel are renormalized and the network is not sensible to spatial correlations between different patches (see also Table 5.1 for a summary of the analytical results).

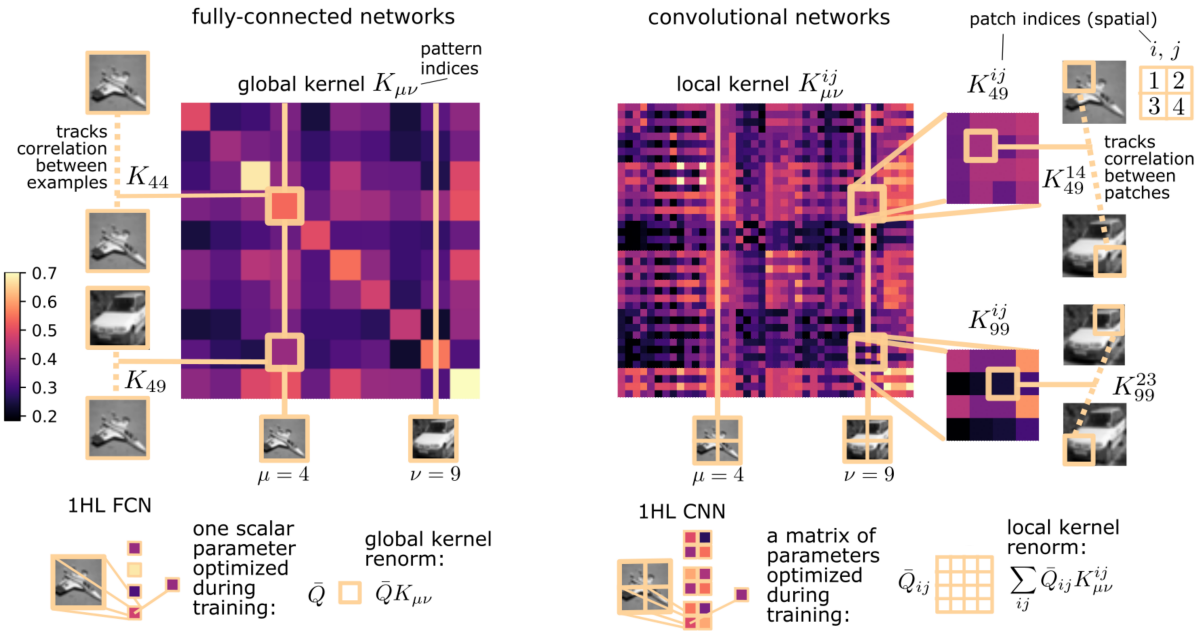


Figure 5.4 –Global and local kernels. Our calculations in the proportional regime reveal a striking difference between FCNs and CNNs. In the first case, the NNGP kernel $K_{\mu\nu}$ that enters in the Bayesian effective action is a global one, which is only capable of tracking global correlations between different training patterns (μ, ν) (left). A FCN network in this regime can fine-tune a single parameter, $\bar{Q}$, that will globally rescale the kernel matrix $K_{\mu\nu}$. On the other hand, in the CNN case we identify a local kernel with two additional (spatial) indices. If we consider $2d$ images, the effect of the local components is to track local correlations between different patches i and j of (possibly) different pair of images (μ, ν) (right). The spatial information contained in these correlations allows the CNN to optimize a matrix of parameters $\bar{Q}_{ij}$, that will renormalize the kernel matrix $K^{ij}_{\mu\nu}$ in a non-trivial way. In the figure, we display the four-index local kernel $K^{ij}_{\mu\nu}$ as a matrix using the multi-index notation $K_{(\mu,i),(\nu,j)}$ and choosing an ordering for the multi-index (μ, i). Global and local kernels are computed for a subset of $P = 10$ gray-scaled CIFAR10 images down-sized to $N_0 = 784$. The local kernels are empirically evaluated for $2d$ convolutions with linear size of the filter mask $M = 14$ and stride $S = 14$ (non-overlapping filters).

5.4 Empirical evidence for global and local kernel renormalization

Extracting the renormalized kernel from the measure of a physical observable is not straightforward and very hard to access in numerical experiments. One indirect experimental blueprint of the form of kernel renormalization at finite width in different architectures is given by the *similarity matrix* of the internal representations before and after training. We define this observable for both FCNs and CNNs as:

$$O_{\mu\nu} \equiv \frac{1}{N_1} \sum_{i=1}^{N_1} \sigma\left(h_i^{\mu}\right) \sigma\left(h_i^{\nu}\right) \,, \tag{5.7}$$

where N_1 denotes the number of neurons in the last layer (for CNNs we have $N_1 = N_c \lfloor N_0/S \rfloor$). We can track the effect of training by taking the difference between the similarity matrix at initialization, which is by definition the NNGP kernel, and the same quantity after training: $\Delta K_{\mu\nu} \equiv \langle O_{\mu\nu} \rangle - K_{\mu\nu}$, where the average is done over the Gibbs ensemble of the weights. An analytical derivation for this observable for FCNs and CNNs, as shown in the Appendix of [106]. The final result in the two cases respectively reads (at zero temperature):

$$\Delta K_{\mu\nu}^{\text{FCN}} = -\frac{1}{N_1} \left[K_{\mu\nu} - \frac{\lambda_1}{\bar{Q}} y^{\mu} y^{\nu} \right] ; \tag{5.8}$$

$$\begin{aligned} \Delta K_{\mu\nu}^{\text{CNN}} = & -\frac{1}{\lambda_1 N_1 \lfloor N_0/S \rfloor} \sum_{ij,\lambda\rho=1}^{\lfloor N_0/S \rfloor, P} \bar{Q}_{ij} P_{\mu\lambda\nu\rho}^{ij} \\ & \times \left[\left(K_{\text{CNN}}^{(\text{R})} \right)_{\lambda\rho}^{-1} - \sum_{\epsilon\omega=1}^{P} \left(K_{\text{CNN}}^{(\text{R})} \right)_{\lambda\epsilon}^{-1} \left(K_{\text{CNN}}^{(\text{R})} \right)_{\rho\omega}^{-1} y^{\epsilon} y^{\omega} \right] , \end{aligned} \tag{5.9}$$

where $P_{\mu\lambda\nu\rho}^{ij} \equiv \frac{1}{2} \sum_k \left[K_{\mu\lambda}^{ik} K_{\nu\rho}^{kj} + K_{\mu\lambda}^{ki} K_{\nu\rho}^{jk} \right]$. Let us now highlight a few interesting physical implications that directly follow from these explicit formulas and can be checked in numerical experiments:

(i) the difference between the trained and untrained similarity matrix $\Delta K_{\mu\nu}^{\text{FCN}}$ converges to zero in the thermodynamic limit $P, N_1 \to \infty$ at fixed $\alpha_1 = P/N_1$, as long as the

terms in the square brackets are of order $o(N_1)$. This means that in the proportional regime under consideration the effect of training on the internal representations is just a finite-size correction $O(1/N_1)$ and the trained similarity matrix stays very close to its value at initialization. In particular, we expect that the empirical distribution of the elements of the matrix $\Delta K^{\text{FCN}}_{\mu\nu}$ will be increasingly peaked around zero in finite-size scaling experiments where we consider increasing values of P and N_1 keeping their ratio $\alpha_1 = P/N_1$ fixed;

(ii) by choosing a learning task with labels $y^\mu = 0, 1$ and input patterns x^μ such that the matrix elements of the kernel $K_{\mu\nu}$ are on average centred in zero, the distribution of the matrix elements of the block $\Delta K^{\text{FCN}}_{\mu\nu}|_{11}$ (corresponding to the subset of training patterns with label one) should drift towards zero at a rate $1/N_1$, if one performs experiments at constant α_1, while increasing P and N_1. Interestingly, the remaining blocks of the matrix $\Delta K^{\text{FCN}}_{\mu\nu}$ should not be affected by this drift;

(iii) the difference between the trained and untrained convolutional similarity matrix $\Delta K^{\text{CNN}}_{\mu\nu}$ displays a different behaviour, due to local kernel renormalization. For instance, even if we restrict our analysis to the learning setting outlined in bullet (ii), the elements of the block $\Delta K^{\text{CNN}}_{\mu\nu}|_{01}$ should not identically converge to zero in the thermodynamic proportional limit. A straightforward scaling analysis in fact shows that the contribution due to the term in square bracket does not vanish in the proportional limit.

The results of a finite-size scaling analysis to check the aforementioned predictions can be found in Fig. 5.5. The results are obtained training 1HL FCNs and CNNs ($1d$ convolutions) on a synthetic dataset composed of random Gaussian patterns whose labels are given by a linear teacher function, i.e. $y = \frac{1}{2}\left[1 + \text{sign}(t \cdot x)\right]$, where t is an N_0-dimensional vector with unitary entries. Overall, the fact that the change in the internal representations of FCNs is a finite-size effect is confirmed by the experiments. On the other hand, the same numerical simulations with CNNs clearly identify a genuine change in the internal representations that occurs also in the thermodynamic proportional limit. We note that such a clean signature of the differences between FCNs and CNNs derive also from our choice of working close to zero

temperature (where the architectures can precisely fit the training data). Equations at finite T are more involved and measuring the indirect effects of global and local kernel renormalization in this case is more difficult. Interestingly, even though our theory should be valid only in the bayesian setting, it turns out to be also predictive the theory worked out here seems to be predictive also for a learning algorithm that does not explicitly sample from the Gibbs posterior distribution. Inde, these experiments were carried out using the so-called ADAM algorithm [57] as learning scheme (see Appendix F for more details).

Another empirical evidence for the local renormalization in CNNs can be found analyzing the bias/variance decomposition of the generalization error over a new example $(\mathbf{x}^0, y^0)$:

$$\langle \epsilon_{\mathrm{g}}(\mathbf{x}^0, y^0) \rangle = (y^0 - \Gamma)^2 + \sigma^2 \tag{5.10}$$

with:

$$\Gamma = \sum_{\mu\nu} \kappa_\mu^{(\mathrm{R})} \left(\frac{\mathbb{1}}{\beta} + K^{(\mathrm{R})}(\{\bar{Q}\}) \right)^{-1}_{\mu\nu} y_\nu, \tag{5.11}$$

$$\sigma^2 = \kappa_0^{(\mathrm{R})} - \sum_{\mu\nu} \kappa_\mu^{(\mathrm{R})} \left(\frac{\mathbb{1}}{\beta} + K^{(\mathrm{R})}(\{\bar{Q}\}) \right)^{-1}_{\mu\nu} \kappa_\nu^{(\mathrm{R})} \tag{5.12}$$

This expressions are valid for FCNs, and can be easily extended to CNNs with the substitution:

$$K^{(\mathrm{R})} \to K^{(\mathrm{R})}_{\mathrm{CNN}} \quad \kappa^{(\mathrm{R})} \to \kappa^{(\mathrm{R})}_{\mathrm{CNN}} \tag{5.13}$$

where the renormalized kernel matrices $K^{(\mathrm{R})}$ and $K^{(\mathrm{R})}_{\mathrm{CNN}}$ are given respectively in Equations (4.8) and (4.54). In the CNN case, these quantities are computed starting from the local covariance matrix defined in Eq. 4.55. These terms undergo the same type of renormalization

as the correspondent kernel matrices (see table 1.1), that is:

$$\kappa_{\mu}^{(\mathrm{R})} = \frac{\bar{Q}}{\lambda_1}\kappa_{\mu}\,, \tag{5.14}$$

$$\left[\kappa_{\mathrm{CNN}}^{(\mathrm{R})}\right]_{\mu} = \frac{1}{\lambda_1 \lfloor N_0/S \rfloor} \sum_{ij=1}^{\lfloor N_0/S \rfloor} \bar{Q}_{ij}\kappa_{\mu}^{ij}\,, \tag{5.15}$$

The $\beta \to \infty$ limit of Eq. (5.11) yields different behaviors for the bias in the two cases:

$$\Gamma^{\mathrm{FC}} = \sum_{\mu\nu} \kappa_{\mu} K^{-1}_{\mu\nu}\, y_{\nu}\,, \tag{5.16}$$

$$\Gamma^{\mathrm{CNN}} = \sum_{\mu\nu} \left[\kappa_{\mathrm{CNN}}^{(\mathrm{R})}\right]_{\mu} \left(K_{\mathrm{CNN}}^{(\mathrm{R})}(\{\bar{Q}\})\right)^{-1}_{\mu\nu}\, y_{\nu}, \tag{5.17}$$

From these expressions one can make two observations: (i) as already discussed in [95], Γ^{FC} at finite width is independent of $\bar{Q}$, and therefore the bias is identical to the infinite-width case for any N_1. (ii) In CNNs, on the other hand, the dependence of the bias from the renormalized Kernel does not disappear, and neither does the dependence on the width N_1.

In the large Gaussian prior regime $\lambda_1 \gg 1$, the previous observations can be made into straightforward predictions that can be tested with numerical experiments. It is easy to check that the variance is vanishing in this regime, since it is always proportional to a term $1/\lambda_1$ both in FCNs and CNNs. Moreover, when $\lambda_1 \gg 1$, the analytical criterion found in [95] to predict whether a finite-width FCN will outperform its infinite-width counterpart, is never satisfied. This is a special case of the general considerations already presented in section 5.2. To summarize, in this regime we expect that (i) the finite FC network can not outperform its infinite-width counterpart: the variance is always decreasing as a function of N_1, while the bias stays constant. (ii) On the contrary, we expect CNNs to possibly beat their infinite-width performance, since the bias can be reduced through the optimization of the feature matrix Q^{ij}.

To test this predictions, we performed numerical simulations with $2d$ CNNs trained on CIFAR10 examples, close to the zero-temperature limit and in the bias-dominated setting where we let the Gaussian prior λ_1 be large. As shown in Fig. 5.6, the test loss of the FCN

is a monotonically decreasing function of the HL size N_1 and it does reach its asymptotic (best) performance already for $\alpha \sim 1$. Compatibly to local kernel renormalization, CNNs with finite number of channels best their infinite-width counterpart, already for large filter sizes ($M = 14, 7$) and ultimately approach the infinite-width limit performance from below. It is worth stressing that we do not expect that local kernel renormalization will lead to improved generalization performance for generic learning tasks, but only in those cases where the training patterns exhibit local spatial correlations, as it occurs in computer vision problems.

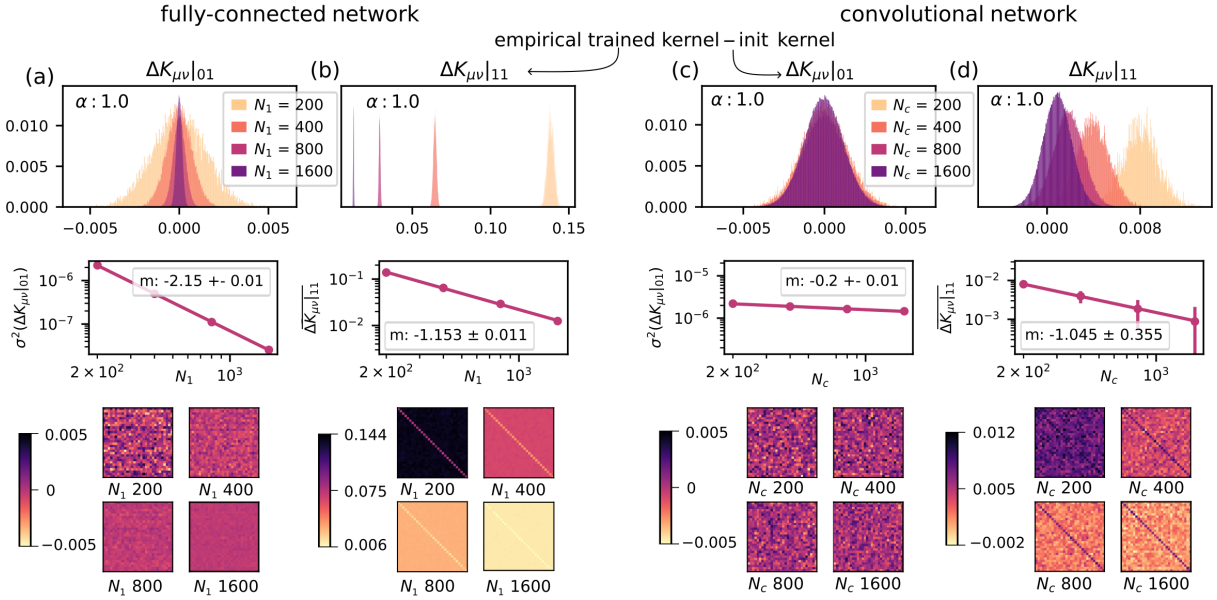


Figure 5.5 –Predicting the effect of global and local kernel renormalization on the internal representations of FCNs and CNNs. A finite-size scaling analysis. Here we show the results of numerical experiments performed in order to check the predictions of Eqs. (5.8), (5.9) for the difference $\Delta K_{\mu\nu}$ between the similarity matrix after and before training. In particular, we train architectures with a single hidden FC/$1d$-convolutional layer with tanh activation (close to zero temperature, see details in Appendix F) on a learning task with random Gaussian inputs and zero/one labels given by a linear teacher. We perform a finite-size scaling analysis by keeping $\alpha_1 = 1$ and choosing increasing values of P, N_1. In panel (a)(top) we observe that the variance of the empirical distribution of the matrix elements of the zero-one block $\Delta K^{\rm FC}_{\mu\nu}|_{01}$ rapidly converges to zero as N_1 grows and α_1 is kept fixed to one, in agreement with the prediction of Eq. (5.8). (middle) In particular the variance goes to zero as $1/N_1^2$. (bottom) The fact that the similarity matrices after and before training are more and more similar in the thermodynamic limit is also clearly visible in the explicit array plot of a subsample of the zero-one block $\Delta K^{\rm FC}_{\mu\nu}|_{01}$. In panel (b)(top) we display the histogram of the distribution of the matrix elements of the one-one block $\Delta K^{\rm FC}_{\mu\nu}|_{11}$. As predicted by Eq. (5.8), the distribution shrinks while (middle) drifting towards zero at a rate $\sim 1/N_1$ and. Overall, (a) and (b) represent a clear indication that the change in the after-training internal representations in 1HL FCNs is a finite-size effect. In (c) and (d) we repeat the same experiment for an architecture with a $1d$-convolutional hidden layer. The striking difference wrt the FC case is that the variance of the empirical distribution of the matrix elements $\Delta K^{\rm CNN}_{\mu\nu}$ does not shrink to zero as N_c grows and α_c is kept fixed to one, as one can also appreciate by directly looking at the array plot of a subsample of the block matrices $\Delta K^{\rm CNN}_{\mu\nu}|_{01}$ and $\Delta K^{\rm CNN}_{\mu\nu}|_{11}$.

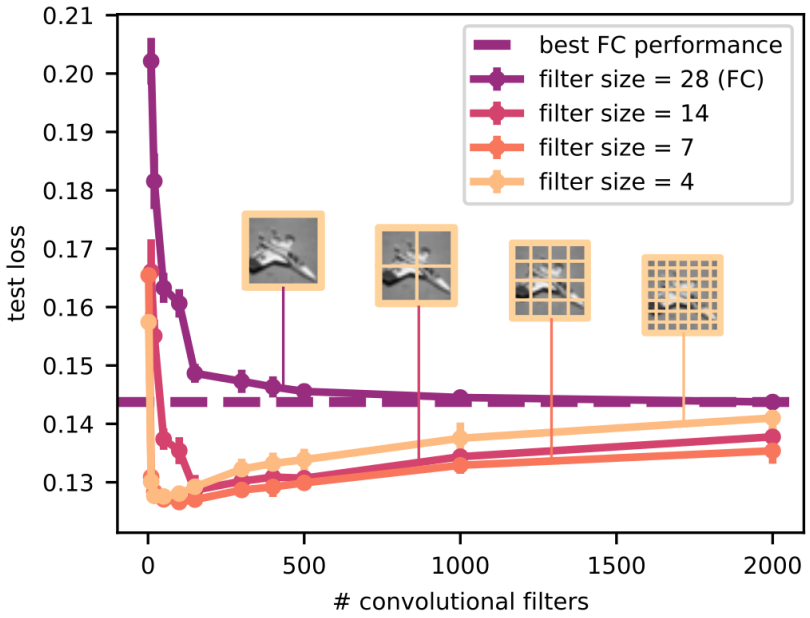


Figure 5.6 –All the networks are trained on the same binary regression task with $P = 500$ examples from two classes of the CIFAR10 dataset, "cars" and "planes", respectively labeled 0 and 1. Error bars represent one standard deviation. The images are grayscaled and coarse-grained to $N_0 = 28 \times 28$. We choose the filter size M to be an integer divisor of 28, and the filters to be non overlapping (taking stride $S = M$). A sketch of how the images are parsed by filters is reported for each network. With this choice, the FC network is retrieved setting $M = 28$. The nets are trained with a zero temperature Langevin dynamics (see appendix REF), and with large Gaussian priors λ_1. In this bias-dominated regime we observe that: (i) in FCNs, the best possible performance is the infinite-width one. Here the variance is monotonically decreasing with the width N_c, while the bias is a constant. Therefore, the test loss approaches its minimum monotonically from above, as predicted by our theory. (ii) finite width CNNs can outperform their infinite-width counterpart. The local kernel renormalization mechanism allows the CNN to optimize both bias and variance at finite width, making it even convenient to have a small number of filters (the minimum is reached for $N_c \sim 50$). Eventually, the CNNs reach their asympthotic performance from below.

Bibliography

[1] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. *Commun. ACM*, 60(6):84–90, may 2017.
→ [piii], [p3]

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.
→ [p]

[3] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.
→ [piii], [p9]

[4] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.
→ [piii], [p35]

[5] Lenka Zdeborová. Understanding deep learning is also a job for physicists. *Nature Physics*, 16(6):602–604, 2020.
→ [p]

[6] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization, 2017.
→ [p]

[7] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning (still) requires rethinking generalization. *Commun. ACM*, 64(3), 2021. → [piii]

[8] Abdulkadir Canatar, Blake Bordelon, and Cengiz Pehlevan. Spectral bias and task-model alignment explain generalization in kernel regression and infinitely wide neural networks. *Nature communications*, 12(1):1–12, 2021. → [piii], [pv], [p36]

[9] Qianyi Li and Haim Sompolinsky. Statistical mechanics of deep linear neural networks: The backpropagating kernel renormalization. *Phys. Rev. X*, 11:031059, Sep 2021. → [p44], [p49], [p57], [p58], [p63], [p67], [p106], [p108], [p109], [p112], [p118], [p139], [p141]

[10] Albert J. Wakhloo, Tamara J. Sussman, and SueYeon Chung. Linear classification of neural manifolds with correlated variability. *Phys. Rev. Lett.*, 131:027301, Jul 2023. → [p]

[11] Inbar Seroussi, Gadi Naveh, and Zohar Ringel. Separation of scales and a thermodynamic description of feature learning in some cnns. *Nature Communications*, 14(1):908, 02 2023. → [p35], [p36], [p39], [p44], [p69], [p139], [p141]

[12] Carlo Baldassi, Clarissa Lauditi, Enrico M. Malatesta, Rosalba Pacelli, Gabriele Perugini, and Riccardo Zecchina. Learning through atypical phase transitions in overparameterized neural networks. *Phys. Rev. E*, 106:014116, Jul 2022. → [p6], [p51]

[13] Yasaman Bahri, Jonathan Kadmon, Jeffrey Pennington, Sam S. Schoenholz, Jascha Sohl-Dickstein, and Surya Ganguli. Statistical mechanics of deep learning. *Annual Review of Condensed Matter Physics*, 11(1):501–528, 2020. → [p]

[14] Alessandro Ingrosso and Sebastian Goldt. Data-driven emergence of convolutional structure in neural networks. *Proceedings of the National Academy of Sciences*, 119(40):e2201854119, 2022. → [p69]

[15] Sebastian Goldt, Marc Mézard, Florent Krzakala, and Lenka Zdeborová. Modeling the influence of data structure on learning in neural networks: The hidden manifold model. *Phys. Rev. X*, 10:041044, Dec 2020. → [piii]

[16] A. Engel and C. Van den Broeck. *Statistical Mechanics of Learning*. Cambridge University Press, 2001. → [piii]

[17] Thomas M. Cover. Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition. *IEEE Transactions on Electronic Computers*, EC-14(3):326–334, 1965. → [p]

[18] E Gardner. Maximum storage capacity in neural networks. *Europhysics Letters (EPL)*, 4(4):481–485, aug 1987. → [p]

[19] E Gardner and B Derrida. Optimal storage properties of neural network models. *Journal of Physics A: Mathematical and General*, 21(1):271–284, jan 1988. → [p]

[20] Rainer Dietrich, Manfred Opper, and Haim Sompolinsky. Statistical mechanics of support vector networks. *Phys. Rev. Lett.*, 82:2975–2978, Apr 1999. → [piii], [pv], [p4], [p36]

[21] Federica Gerace, Bruno Loureiro, Florent Krzakala, Marc Mézard, and Lenka Zdeborová. Generalisation error in learning with random features and the hidden manifold model. *Journal of Statistical Mechanics: Theory and Experiment*, 2021(12):124013, dec 2021. → [piii]

[22] Sebastian Goldt, Bruno Loureiro, Galen Reeves, Florent Krzakala, Marc Mezard, and Lenka Zdeborová. The gaussian equivalence of generative models for learning with shallow neural networks. In *Proceedings of the 2nd Mathematical and Scientific Machine Learning Conference*, volume 145 of *Proceedings of Machine Learning Research*, pages 426–471. 2nd Mathematical and Scientific Machine Learning Conference, 2021. Online, August 16-19, 2021. → [p]

[23] Hugo Cui, Florent Krzakala, and Lenka Zdeborová. Optimal learning of deep random networks of extensive-width. *arXiv preprint arXiv:2302.00375*, 2023. → [piii], [p44]

[24] Radford M. Neal. *Priors for Infinite Networks*, pages 29–53. Springer New York, New York, NY, 1996. → [piv], [p14], [p17], [p19], [p23], [p24]

[25] Christopher Williams. Computing with infinite networks. In M.C. Mozer, M. Jordan, and T. Petsche, editors, *Advances in Neural Information Processing Systems*, volume 9. MIT Press, 1996. → [p23]

[26] Alexander G. de G. Matthews, Jiri Hron, Mark Rowland, Richard E. Turner, and Zoubin Ghahramani. Gaussian process behaviour in wide deep neural networks. In *International Conference on Learning Representations*, 2018. → [p23], [p24], [p43]

[27] Jaehoon Lee, Jascha Sohl-dickstein, Jeffrey Pennington, Roman Novak, Sam Schoenholz, and Yasaman Bahri. Deep neural networks as gaussian processes. In *International Conference on Learning Representations*, 2018.
→ [p19], [p25], [p29], [p38], [p43], [p46], [p50], [p67], [p69], [p109]

[28] Adrià Garriga-Alonso, Carl Edward Rasmussen, and Laurence Aitchison. Deep convolutional networks as shallow gaussian processes. In *International Conference on Learning Representations*, 2019. → [p]

[29] Roman Novak, Lechao Xiao, Yasaman Bahri, Jaehoon Lee, Greg Yang, Daniel A. Abolafia, Jeffrey Pennington, and Jascha Sohl-dickstein. Bayesian deep convolutional networks with many channels are gaussian processes. In *International Conference on Learning Representations*, 2019. → [pv], [p23], [p30], [p31], [p37], [p39], [p43], [p59], [p62], [p69], [p131]

[30] Arthur Jacot, Franck Gabriel, and Clement Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. → [p32], [p33], [p43]

[31] Lénaïc Chizat, Edouard Oyallon, and Francis Bach. On lazy training in differentiable programming. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. → [p23]

[32] Jaehoon Lee, Lechao Xiao, Samuel Schoenholz, Yasaman Bahri, Roman Novak, Jascha Sohl-Dickstein, and Jeffrey Pennington. Wide neural networks of any depth evolve as linear models under gradient descent. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. → [piv], [p23], [p29], [p32], [p37]

[33] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, Sep 1995. → [pv], [p3], [p5], [p6], [p36]

[34] Blake Bordelon, Abdulkadir Canatar, and Cengiz Pehlevan. Spectrum dependent learning curves in kernel regression and wide neural networks. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 1024–1034. PMLR, 13–18 Jul 2020. → [pv], [p36]

[35] Jiri Hron, Yasaman Bahri, Jascha Sohl-Dickstein, and Roman Novak. Infinite attention: NNGP and NTK for deep attention networks. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 4376–4386. PMLR, 13–18 Jul 2020. → [pv], [p23], [p43]

[36] Greg Yang and Edward J. Hu. Tensor programs iv: Feature learning in infinite-width neural networks. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 11727–11737. PMLR, 18–24 Jul 2021. → [pv], [p23]

[37] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828, 2013. → [pv], [p35]

[38] Yu Han Liu. Feature extraction and image recognition with convolutional neural networks. *Journal of Physics: Conference Series*, 1087(6):062032, sep 2018. → [p35]

[39] Jianchang Mao and A.K. Jain. Artificial neural networks for feature extraction and multivariate data projection. *IEEE Transactions on Neural Networks*, 6(2):296–317, 1995. → [p35]

[40] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828, 2013. → [p]

[41] Dong Yu, Michael L Seltzer, Jinyu Li, Jui-Ting Huang, and Frank Seide. Feature learning in deep neural networks-studies on speech recognition tasks. *arXiv preprint arXiv:1301.3605*, 2013. → [pv]

[42] Nikhil Vyas, Yamini Bansal, and Nakkiran Preetum. Limitations of the ntk for understanding generalization in deep learning. *arXiv preprint arXiv:2206.10012*, 2022. → [pv], [p36]

[43] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015. → [p3], [p9]

[44] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C.J. Burges, L. Bottou, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012. → [p3]

[45] Dae-Young Kang, Pham Duong, and Jung-Chul Park. Application of deep learning in dentistry and implantology. *The Korean Academy of Oral and Maxillofacial Implantology*, 24:148–181, 09 2020. → [p4]

[46] E Gardner and B Derrida. Optimal storage properties of neural network models. *Journal of Physics A: Mathematical and General*, 21(1):271, jan 1988. → [p6]

[47] Carlo Baldassi, Clarissa Lauditi, Enrico M. Malatesta, Gabriele Perugini, and Riccardo Zecchina. Unveiling the structure of wide flat minima in neural networks. *Phys. Rev. Lett.*, 127:278301, Dec 2021. → [p51]

[48] Carlo Baldassi, Fabrizio Pittorino, and Riccardo Zecchina. Shaping the learning landscape in neural networks around wide flat minima. *Proceedings of the National Academy of Sciences*, 117:201908636, 12 2019. → [p6], [p13], [p51]

[49] Frank Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6):386–408, 1958. → [p8]

[50] S. Linnainmaa. The representation of the cumulative rounding error of an algorithm as a taylor expansion of the local rounding errors. Master's thesis, Univ. Helsinki, 1970. See chapters 6-7 and FORTRAN code on pages 58-60. [PDF Link]. The first publication on "modern" backpropagation, also known as the reverse mode of automatic differentiation. See also BIT 16, 146-160, 1976. [Link]. → [p9]

[51] P. J. Werbos. Applications of advances in nonlinear sensitivity analysis. In R. Drenick and F. Kozin, editors, *System Modeling and Optimization: Proc. IFIP*. Springer, 1982. First application of backpropagation[BP1] to NNs (concretizing thoughts in his 1974 thesis). [PDF Link]. → [p9]

[52] D. H. Hubel and T. N. Wiesel. Receptive fields of single neurones in the cat's striate cortex. *The Journal of Physiology*, 148(3):574–591, 1959. → [p10]

[53] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1(4):541–551, 1989. → [p10]

[54] Francesca Mignacco, Pierfrancesco Urbani, and Lenka Zdeborová. Stochasticity helps to navigate rough landscapes: comparing gradient-descent-based algorithms in the phase retrieval problem. *Machine Learning: Science and Technology*, 2(3):035029, jul 2021. → [p13]

[55] Carlo Baldassi, Alessandro Ingrosso, Carlo Lucibello, Luca Saglietti, and Riccardo Zecchina. Local entropy as a measure for sampling solutions in constraint satisfaction problems. *Journal of Statistical Mechanics: Theory and Experiment*, 2016(2):023301, feb 2016. → [p13], [p51]

[56] Léon Bottou, Frank E. Curtis, and Jorge Nocedal. Optimization methods for large-scale machine learning, 2016. quantization overview. → [p13]

[57] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. → [p75]

[58] Pratik Chaudhari, Anna Choromanska, Stefano Soatto, Yann LeCun, Carlo Baldassi, Christian Borgs, Jennifer Chayes, Levent Sagun, and Riccardo Zecchina. Entropy-SGD: Biasing gradient descent into wide valleys. In *International Conference on Learning Representations*, 2017. → [p13], [p51]

[59] Leslie N. Smith and Nicholay Topin. Super-convergence: very fast training of neural networks using large learning rates. In Tien Pham, editor, *Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications*, volume 11006, page 1100612. International Society for Optics and Photonics, SPIE, 2019. → [p13]

[60] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 1026–1034, 2015. → [p13]

[61] Greg Yang and Edward J. Hu. Tensor programs iv: Feature learning in infinite-width neural networks. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 11727–11737. PMLR, 18–24 Jul 2021. → [p13]

[62] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 2005. → [p20]

[63] Greg Yang. Tensor programs i: Wide feedforward or recurrent neural networks of any architecture are gaussian processes, 2021. → [p23], [p43]

[64] Song Mei, Andrea Montanari, and Phan-Minh Nguyen. A mean field view of the landscape of two-layer neural networks. *Proceedings of the National Academy of Sciences*, 115(33):E7665–E7671, 2018. → [p23], [p40], [p69]

[65] Justin Sirignano and Konstantinos Spiliopoulos. Mean field analysis of neural networks: A central limit theorem. *Stochastic Processes and their Applications*, 130(3):1820–1852, March 2020. Funding Information: K.S. was partially supported by the National Science Foundation, United States (DMS 1550918). Publisher Copyright: © 2019 Elsevier B.V. → [p]

[66] Justin Sirignano and Konstantinos Spiliopoulos. Mean field analysis of deep neural networks. *Math. Oper. Res.*, 47(1):120–152, feb 2022. → [p]

[67] Jiahui Yu and Konstantinos Spiliopoulos. Normalization effects on shallow neural networks and related asymptotic expansions. *Foundations of Data Science*, 3(2):151–200, 2021. → [p]

[68] Grant Rotskoff and Eric Vanden-Eijnden. Trainability and accuracy of artificial neural networks: An interacting particle system approach. *Communications on Pure and Applied Mathematics*, 75(9):1889–1935, jul 2022. → [p23]

[69] Boris Hanin. Neural networks and large width and depth. 2023. → [p24], [p32]

[70] B. E. Fristedt and L. F. Gray. *A modern approach to probability theory*. Birkhäuser, Boston, 1996. Theorems 14.15 and 18.21. → [p26]

[71] Boris Hanin. Correlation functions in random fully connected neural networks at finite width, 04 2022. → [p26]

[72] Jaehoon Lee, Samuel Schoenholz, Jeffrey Pennington, Ben Adlam, Lechao Xiao, Roman Novak, and Jascha Sohl-Dickstein. Finite versus infinite neural networks: an empirical study. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 15156–15172. Curran Associates, Inc., 2020. → [p29], [p37], [p69]

[73] Simon S. Du, Jason D. Lee, Yuandong Tian, Aarti Singh, and Barnabás Póczos. Gradient descent learns one-hidden-layer CNN: don't be afraid of spurious local minima. In Jennifer G. Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 1338–1347. PMLR, 2018. → [p32], [p33]

[74] Chaoyue Liu, Libin Zhu, and Mikhail Belkin. Toward a theory of optimization for over-parameterized systems of non-linear equations: the lessons of deep learning. *ArXiv*, abs/2003.00307, 2020. → [p32]

[75] Lenaic Chizat, Edouard Oyallon, and Francis Bach. On lazy training in differentiable programming, 2020. → [p33]

[76] Yoshua Bengio, Aaron C Courville, and Pascal Vincent. Unsupervised feature learning and deep learning: A review and new perspectives. *CoRR, abs/1206.5538*, 1(2665):2012, 2012. → [p35]

[77] Yandong Wen, Kaipeng Zhang, Zhifeng Li, and Yu Qiao. A discriminative feature learning approach for deep face recognition. volume 9911, pages 499–515, 10 2016. → [p35]

[78] Gadi Naveh and Zohar Ringel. A self consistent theory of gaussian processes captures feature learning effects in finite cnns. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 21352–21364. Curran Associates, Inc., 2021. → [p35], [p39], [p44]

[79] Jacob Zavatone-Veth, Abdulkadir Canatar, Ben Ruben, and Cengiz Pehlevan. Asymptotics of representation learning in finite bayesian neural networks. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 24765–24777. Curran Associates, Inc., 2021. → [p]

[80] Jacob A. Zavatone-Veth, William L. Tong, and Cengiz Pehlevan. Contrasting random and learned features in deep bayesian linear regression. *Phys. Rev. E*, 105:064118, 06 2022. → [p63], [p67]

[81] Daniel A. Roberts, Sho Yaida, and Boris Hanin. *The Principles of Deep Learning Theory*. Cambridge University Press, 2022. `https://deeplearningtheory.com`. → [p]

[82] Boris Hanin. Random fully connected neural networks as perturbatively solvable hierarchies, 2023. → [p]

[83] Joseph M. Antognini. Finite size corrections for neural network gaussian processes, 2019. → [p]

[84] Sho Yaida. Non-Gaussian processes and neural networks at finite widths. In Jianfeng Lu and Rachel Ward, editors, *Proceedings of The First Mathematical and Scientific Machine Learning Conference*, volume 107 of *Proceedings of Machine Learning Research*, pages 165–192. PMLR, 07 2020. → [p]

[85] Laurence Aitchison. Why bigger is not always better: on finite and infinite neural networks. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 156–164. PMLR, 07 2020. → [p51], [p116]

[86] Adam X. Yang, Maxime Robeyns, Edward Milsom, Nandi Schoots, and Laurence Aitchison. A theory of representation learning in deep neural networks gives a deep generalisation of kernel methods, 2023. → [p35], [p36], [p39], [p51], [p116]

[87] Geoff Pleiss and John Patrick Cunningham. The limitations of large width in neural networks: A deep gaussian process perspective. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021. → [p36]

[88] Yu hsin Chen, Ignacio Lopez Moreno, Tara Sainath, Mirkó Visontai, Raziel Alvarez, and Carolina Parada. Locally-connected and convolutional neural networks for small footprint speaker recognition. In *Interspeech*, 2015. → [p37]

[89] Alexander Atanasov, Blake Bordelon, Sabarish Sainathan, and Cengiz Pehlevan. The onset of variance-limited behavior for networks in the lazy and rich regimes. *arXiv preprint arXiv:2212.12147*, 2022. → [p37], [p38], [p69]

[90] Francesco Cagnetta, Alessandro Favero, and Matthieu Wyart. What can be learnt with wide convolutional neural networks?, 2023. → [p39]

[91] Alessandro Favero, Francesco Cagnetta, and Matthieu Wyart. Locality defeats the curse of dimensionality in convolutional teacher-student scenarios. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021. → [p39]

[92] Leonardo Petrini, Francesco Cagnetta, Eric Vanden-Eijnden, and Matthieu Wyart. Learning sparse features can lead to overfitting in neural networks. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022. → [p40]

[93] Mario Geiger, Leonardo Petrini, and Matthieu Wyart. Landscape and training regimes in deep learning. *Physics Reports*, 924:1–18, 2021. Landscape and training regimes in deep learning. → [p40]

[94] Mario Geiger, Stefano Spigler, Arthur Jacot, and Matthieu Wyart. Disentangling feature and lazy training in deep neural networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2020(11):113301, nov 2020. → [p40], [p69]

[95] R. Pacelli, S. Ariosto, M. Pastore, F. Ginelli, M. Gherardi, and P. Rotondo. A statistical mechanics framework for bayesian deep neural networks beyond the infinite-width limit. 2023. → [p40], [p45], [p46], [p53], [p55], [p58], [p61], [p68], [p76], [p133], [p134], [p141]

[96] Qianyi Li and Haim Sompolinsky. Globally gated deep linear networks. *arXiv preprint arXiv:2210.17449*, 2022. → [p44], [p106]

[97] Dominik Schröder, Hugo Cui, Daniil Dmitriev, and Bruno Loureiro. Deterministic equivalent and error universality of deep random features learning. *arXiv preprint arXiv:2302.00401*, 2023. → [p44]

[98] Boris Hanin and Alexander Zlokapa. Bayesian interpolation with deep linear networks. *Proceedings of the National Academy of Sciences*, 120(23):e2301345120, 2023. → [p44], [p49], [p100], [p106], [p112]

[99] Péter Breuer and Péter Major. Central limit theorems for non-linear functionals of gaussian fields. *Journal of Multivariate Analysis*, 13(3):425–441, 1983.
→ [p46], [p48], [p99], [p105], [p123], [p126], [p133]

[100] Amar Shah, Andrew Wilson, and Zoubin Ghahramani. Student-t Processes as Alternatives to Gaussian Processes. In Samuel Kaski and Jukka Corander, editors, *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics*, volume 33 of *Proceedings of Machine Learning Research*, pages 877–885, Reykjavik, Iceland, 22–25 Apr 2014. PMLR. → [p51]

[101] A C C Coolen, M Sheikh, A Mozeika, F Aguirre-López, and F Antenucci. Replica analysis of overfitting in generalized linear regression models. *Journal of Physics A: Mathematical and Theoretical*, 53(36):365001, aug 2020. → [p]

[102] Alexander Mozeika, Mansoor Sheikh, Fabian Aguirre-López, Fabrizio Antenucci, and Anthony C. C. Coolen. Exact results on high-dimensional linear regression via statistical physics. *Phys. Rev. E*, 103:042142, Apr 2021. → [p]

[103] Yusuke Uchiyama, Hiroki Oka, and Ayumu Nono. Student's t-process regression on the space of probability density functions. *Proceedings of the ISCIE International Symposium on Stochastic Systems Theory and its Applications*, 2021:1–5, 2021. → [p51]

[104] Hyungi Lee, Eunggu Yun, Hongseok Yang, and Juho Lee. Scale mixtures of neural network gaussian processes. In *International Conference on Learning Representations*, 2022. → [p51]

[105] Jacob A. Zavatone-Veth and Cengiz Pehlevan. Depth induces scale-averaging in overparameterized linear bayesian neural networks. In *2021 55th Asilomar Conference on Signals, Systems, and Computers*, pages 600–607, 2021. → [p51], [p116]

[106] R. Aiudi, R. Pacelli, A. Vezzani, R. Burioni, and P. Rotondo. Local kernel renormalization as a mechanism for feature learning in overparametrized convolutional neural networks, 2023. → [p61], [p73]

[107] Jacob Zavatone-Veth, Abdulkadir Canatar, Ben Ruben, and Cengiz Pehlevan. Asymptotics of representation learning in finite bayesian neural networks. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 24765–24777. Curran Associates, Inc., 2021. → [p63], [p67]

[108] Jacob A Zavatone-Veth, Abdulkadir Canatar, Benjamin S Ruben, and Cengiz Pehlevan. Asymptotics of representation learning in finite bayesian neural networks*. *Journal of Statistical Mechanics: Theory and Experiment*, 2022(11):114008, 11 2022. → [p63], [p67]

[109] Youngmin Cho and Lawrence Saul. Kernel methods for deep learning. In Y. Bengio, D. Schuurmans, J. Lafferty, C. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems*, volume 22. Curran Associates, Inc., 2009. → [p67], [p140]

[110] Ben Poole, Subhaneil Lahiri, Maithra Raghu, Jascha Sohl-Dickstein, and Surya Ganguli. Exponential expressivity in deep neural networks through transient chaos. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016. → [p67]

[111] Ge Yang and Samuel Schoenholz. Mean field residual networks: On the edge of chaos. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. → [p67]

[112] Brendan D. Tracey and David Wolpert. *Upgrading from Gaussian Processes to Student's-T Processes.* → [p67]

[113] Ivan Nourdin, Giovanni Peccati, and Mark Podolskij. Quantitative Breuer-Major theorems, 2010. → [p48], [p98], [p105]

[114] Jean-Marc Bardet and Donatas Surgailis. Moment bounds and central limit theorems for gaussian subordinated arrays. *Journal of Multivariate Analysis*, 114:457–473, 2013. → [p48], [p99], [p105]

[115] V A Marčenko and L A Pastur. Distribution of eigenvalues for some sets of random matrices. *Mathematics of the USSR-Sbornik*, 1(4):457, apr 1967. → [p114]

[116] P.J. Forrester. *Log-Gases and Random Matrices (LMS-34)*. London Mathematical Society Monographs. Princeton University Press, 2010. → [p114]

[117] Guofei Pang, Liu Yang, and George Em Karniadakis. Neural-net-induced gaussian process regression for function approximation and pde solution. *Journal of Computational Physics*, 384:270–288, 2019. → [p140]

[118] Rosalba Pacelli. rpacelli/FC_deep_bayesian_networks: FC_deep_bayesian_networks, August 2023. → [p140]

APPENDIX A

Technical remarks on the validity of the framework

A.1 The Breuer-Major theorem as a justification for the Gaussian equivalence in shallow networks

The Breuer-Major theorem and its extensions deal with the following sequence of random variables:

$$S_N = \frac{1}{\sqrt{N}} \sum_{i=1}^{N} c_i F(x_i) \quad N \geq 1 \,. \tag{A.1}$$

Clearly, if the distribution of the vector $\mathbf{x} = (x_1, \ldots, x_N)$ is factorized over its coordinates, i.e. $p(\mathbf{x}) = \prod_i p(x_i)$ and $F(x) = x$, the random variable $S = \lim_{N\to\infty} S_N$ is normal distributed as long as the mean $\mathbb{E}(x_i) = 0$, the variance $\mathbb{E}(x_i^2)$ is finite and the c_i's satisfy the so-called Lindeberg's condition. This is also true whenever F is a well-behaved non-linearity.

The Breuer-Major theorem essentially extends this result to generic GPs, providing sufficient conditions on the covariance matrix of the GP and on the non-linearity F that guarantee

convergence of S_N to the normal distribution. We report here the modern statement of the theorem given in Ref. [113].

As a first step, consider a stationary (unidimensional) GP $x = (x_k)_{k\in\mathbb{Z}}$. Stationarity –which is not essential and will be replaced by a weaker condition in the following– amounts to require that the covariance of the process $C_{ij} = \mathbb{E}(x_i x_j)$ is a function of the difference $i - j$, i.e. $C_{ij} = C(i - j)$. The only technical condition to be imposed on the non-linear function F is to have well-defined *Hermite rank* R. The Hermite rank is the smallest positive integer that appears in the decomposition of F over the Hermite polynomials:

$$F(x) = \sum_{k=R}^{\infty} f_k \, \mathrm{He}_k(x) \,, \tag{A.2}$$

where $\mathrm{He}_k(x)$ is the k-th Hermite polynomial and f_k the coefficient of the expansion. For many reasonable activation functions F, $R = 1$.

Theorem 3 (Breuer and Major, 1983)**.** *Let $x = (x_k)_{k\in\mathbb{Z}}$ be a stationary unidimensional GP with covariance $C(i - j)$. Let $\mathbb{E}\left[F(x_1)\right] = 0$ and $\mathbb{E}\left[F^2(x_1)\right] < \infty$ and assume that the function F has Hermite rank $R \geq 1$. Suppose that:*

$$\sum_{j\in\mathbb{Z}} |C_{1j}|^R < \infty \,. \tag{A.3}$$

Then $\sigma^2 := \mathbb{E}\left[F(x_1)^2\right] + 2\sum_{j=1}^{\infty} \mathbb{E}\left[F(x_1)F(x_j)\right]$ is finite. Moreover, one has that the sequence of random variables

$$S_N = \frac{1}{\sqrt{N}} \sum_{i=1}^{N} F(x_i) \quad N \geq 1 \tag{A.4}$$

converges in distribution to $\mathcal{N}(0, \sigma^2)$, i.e. to a Gaussian distribution with zero mean and variance σ^2.

For our scopes we will need a slightly stronger statement than the one just mentioned: (i) in our calculation the covariance will not be stationary and (ii) we will need to consider a

more general sequence of nonlinear functions $c_i F(x_i)$, such that each term of the sum (A.4) is weighted by a factor $c_i \neq 1$.

It has been shown, already in the original reference [99], that the hypothesis of stationarity can be weakened and replaced with a requirement of uniform convergence of the elements of the covariance, namely:

$$\sum_{j\in\mathbb{Z}} |C_{ij}|^R < B_0 \quad \forall i \in \mathbb{Z}\,, \tag{A.5}$$

where B_0 is a positive finite constant. Extension (ii) has been addressed more recently in [114] under mild technical assumptions.

A.2 Constraints on the scaling of the size of the dataset P with the input dimension N_0

In this section I address the additional constraints to the thermodynamic scaling ($P, N_1 \to \infty$ with $\alpha_1 = P/N_1$ finite) that may come from the hypotheses of the Breuer-Major on the covariance matrix C. The only stringent condition to verify is equation (A.5), that is

$$\sum_{\mu=1}^{P} |C_{\mu\nu}|^R < B_0 \qquad \forall\, \nu = 1, \dots, P\,, \tag{A.6}$$

where B_0 is a given finite constant and R the Hermite rank of the activation function σ. In the case of inputs $\mathbf{x}$ with i.i.d. standard Gaussian coordinates, $C_{\mu\nu}$ is a Wishart random matrix with off-diagonal entries of order $1/\sqrt{N_0}$ and random signs: after taking the absolute value, the sum in Eq (A.6) is of order $P(N_0)^{-R/2}$. Note that this provides an infinite class of activation functions (those with Hermite rank $R \geq 2$) where we can safely work at least at finite $\alpha_0 = P/N_0$. For activation functions with Hermite rank $R = 1$ (such as Erf or ReLU) we cannot provide such a guarantee by only looking at the hypothesis of the Breuer-Major theorem. It is also worth stressing that, given any odd (non-odd) activation function $\sigma(x)$ with Hermite rank

$R = 1$, it is easy to engineer a new reasonable activation function with Hermite rank $R = 3$ ($R = 2$), just by replacing the old activation function with a new one $\sigma_1(x) = \sigma(x) - g_1 x$, where the coefficient $g_1 = \langle \sigma(x)\, \mathrm{He}_1(x) \rangle$ and the average is over a normal distribution of zero mean and unit variance.

Observe that there is at least one case of activation function with $R = 1$ where the derivation goes through at finite α_0, i.e. the linear function $\sigma(x) = x$ (in this case we can obtain the result at finite P, N_1, N_0, as done also in Ref. [98]). In the supplemental material, Sec. C.1, we examine the specific case of quadratic activation $\sigma(x) = x + x^2$ (that has $R = 1$), deriving the final effective action without employing the BM theorem. As in the linear case, this derivation goes through at finite α_0. We are thus led to think that the scaling $P = O(\sqrt{N_0})$ suggested for $R = 1$ is overly-pessimistic.

APPENDIX B

Details of calculations for fully-connected networks in the proportional regime

B.1 Computation of Kernel elements

The NNGP kernel, evaluated for a couple of examples $\mathbf{x}^\mu, \mathbf{x}^\nu$ reads:

$$K(\mathbf{x}^\mu, \mathbf{x}^\nu) = \int \mathrm{d}\{\mathbf{h}\} p(\{\mathbf{h}\}) \sigma(h(\mathbf{x}^\mu)) \sigma(h(\mathbf{x}^\nu)), \tag{B.1}$$

$$p(\{\mathbf{h}\}) = \mathcal{N}_{\mathbf{h}}(0, C) \equiv \frac{e^{\sum_{\mu\nu} h^\mu C^{-1}_{\mu\nu} h^\nu}}{\sqrt{2\pi \det C}}, \qquad C_{\mu\nu} = \frac{\mathbf{x}_\mu \cdot \mathbf{x}_\nu}{\lambda_0 N_0}. \tag{B.2}$$

To reduce the integral from a P dimensional to a 2 dimensional one we need to assume that the (anti)fourier transform of the activation function exists:

$$\sigma(h) = \int e^{ikh} \hat{\sigma}(k) dk \tag{B.3}$$

We denote $\mathbf{h}$ a vector of components h^μ, and $\boldsymbol{k}$ a vector of the same size that has ones in position (μ, ν), and 0 elsewhere. We can rewrite K:

$$K_{\mu\nu} = \int dk^1 dk^2 \int dp\left(\{\mathbf{h}\}\right)\ \hat{\sigma}(k^1)\hat{\sigma}(k^2) e^{i\boldsymbol{k}\cdot\mathbf{h}} \tag{B.4}$$

$$p\left(\{\mathbf{h}\}\right) = \frac{1}{\sqrt{(2\pi)^{PY}\det\Gamma}}\, e^{-\frac{1}{2}\sum_{\alpha,\beta}^{Y\ P}{}_{\mu\nu=0} h^{\mu\alpha}(C^{-1})^{\alpha\beta}_{\mu\nu} h^{\nu\beta}} \tag{B.5}$$

Defining

$$\tilde{C} = \begin{pmatrix} C_{\mu\mu} & C_{\mu\nu} \\ C^1_{\mu\nu} & C_{\nu\nu}, \end{pmatrix} \tag{B.6}$$

we can perform the $\mathbf{h}$ integrals and rewrite:

$$\begin{aligned} K_{\mu\nu} &= \int dk^1 dk^2 \int \frac{1}{\sqrt{(2\pi)^2 \det\tilde{C}}}\, e^{-\frac{1}{2}\begin{pmatrix} k^1 & k^2 \end{pmatrix}\begin{pmatrix} C_{\mu\mu} & C_{\mu\nu} \\ C_{\mu\nu} & C_{\nu\nu} \end{pmatrix}\begin{pmatrix} k^1 \\ k^2 \end{pmatrix}} \hat{\sigma}(k^1)\hat{\sigma}(k^2) \\ &= \int \mathrm{d}\mathbf{t} \frac{1}{\sqrt{(2\pi)^2 \det\tilde{C}}}\, e^{-\frac{1}{2}\mathbf{t}^T\left[\tilde{C}\right]^{-1}\mathbf{t}} \sigma(t^1)\sigma(t^2) \end{aligned} \tag{B.7}$$

where I used the notation $\mathbf{t} \equiv (t_1, t_2)$.

Note on the same derivation in the replicated ensemble

We have now that $\mathbf{h}$ is a vector of components $h^{\mu\alpha}$, and $\boldsymbol{k}$ a vector of the same size that has ones in position $(\mu\alpha, \nu\alpha)$, and 0 elsewhere. With the same assumption on the anti-fourier transform we can rewrite B.10:

$$K^1_{\mu\nu} = \int dk^1 dk^2 \int dp\left(\{\mathbf{h}\}\right)\ \hat{\sigma}(k^1)\hat{\sigma}(k^2) e^{i\boldsymbol{k}\cdot\mathbf{h}} \tag{B.8}$$

$$p\left(\{\mathbf{h}\}\right) = \frac{1}{\sqrt{(2\pi)^{PY}\det\Gamma}}\, e^{-\frac{1}{2}\sum_{\alpha,\beta}^{Y\ P}{}_{\mu\nu=0} h^{\mu\alpha}(\Gamma^{-1})^{\alpha\beta}_{\mu\nu} h^{\nu\beta}} \tag{B.9}$$

Now we can perform the h integrals:

$$K^1_{\mu\nu} = \int dk^1 dk^2 \int \frac{1}{\sqrt{(2\pi)^2 \det \tilde{\Gamma}}} e^{-\frac{1}{2}\begin{pmatrix} k^1 & k^2 \end{pmatrix}\begin{pmatrix} \Gamma^1_{\mu\mu} & \Gamma^1_{\mu\nu} \\ \Gamma^1_{\mu\nu} & \Gamma^1_{\nu\nu} \end{pmatrix}\begin{pmatrix} k^1 \\ k^2 \end{pmatrix}} \hat{\sigma}(k^1)\hat{\sigma}(k^2)$$
$$= \int \frac{1}{\sqrt{(2\pi)^2 \det \tilde{\Gamma^1}}} e^{-\frac{1}{2}\begin{pmatrix} t^1 & t^2 \end{pmatrix}\begin{pmatrix} \Gamma^1_{\mu\mu} & \Gamma^1_{\mu\nu} \\ \Gamma^1_{\mu\nu} & \Gamma^1_{\nu\nu} \end{pmatrix}^{-1}\begin{pmatrix} t^1 \\ t^2 \end{pmatrix}} \sigma(t^1)\sigma(t^2) \tag{B.10}$$

with the proper normalisation. Analogously one can find the expression for K^0:

$$K^0_{\mu\nu} = \int dt^1 dt^2 \int \frac{1}{\sqrt{(2\pi)^2 \det \tilde{\Gamma}}} e^{-\frac{1}{2}\begin{pmatrix} t^1 & t^2 \end{pmatrix}\begin{pmatrix} \Gamma^1_{\mu\mu} & \Gamma^0_{\mu\nu} \\ \Gamma^0_{\mu\nu} & \Gamma^1_{\nu\nu} \end{pmatrix}^{-1}\begin{pmatrix} t^1 \\ t^2 \end{pmatrix}} \sigma(t^1)\sigma(t^2) \tag{B.11}$$

with Γ_1 and Γ_0 defined in .

B.2 Effective action for a 1HL network in the proportional regime

In this section I will discuss the details of the calculation of the 1HL partition function in the proportional regime. The starting point is the following partition function:

$$Z = \int \prod_{i_1}^{N_1} \mathrm{d}v_{i_1} \prod_{i_1,i_0}^{N_1,N_0} \mathrm{d}w_{i_1 i_0} \exp\Bigg\{ -\frac{\lambda_1}{2}\sum_{i_1}^{N_1} v_{i_1}^2 - \frac{\lambda_0}{2}\|w\|^2 - \frac{\beta}{2}\sum_{\mu}^{P}\left[y^\mu - \frac{1}{\sqrt{N_1}}\sum_{i_1}^{N_1} v_{i_1}\sigma\left(\sum_{i_0}^{N_0}\frac{w_{i_1,i_0}x^\mu_{i_0}}{\sqrt{N_0}}\right)\right]^2 \Bigg\}. \tag{B.12}$$

where $w = W^{(1)}$ and we took $b^{(1)} = 0$ without loss generality [1]. The first step is to decouple the weights of the different layers in the loss function. This can be done including standard identities built over two families of Dirac deltas, one for the pre-activations of the hidden layer and one for the output of the network:

$$1 = \int \prod_{\mu}^{P} ds^{\mu} \delta \left[s^{\mu} - \frac{1}{\sqrt{N_1}} \sum_{i}^{N_1} v_{i_1} \sigma(h_{i_1}^{\mu}) \right] , \tag{B.13}$$

$$1 = \int \prod_{\mu}^{P} \prod_{i_1}^{N_1} dh_{i_1}^{\mu} \, \delta \left(h_{i_1}^{\mu} - \frac{1}{\sqrt{N_0}} \sum_{i_0}^{N_0} w_{i_1 i_0} x_{i_0}^{\mu} \right) . \tag{B.14}$$

By using a standard Fourier representation of these deltas, which introduces the conjugate variables $\bar{h}_{i_1}^{\mu}$ and $\bar{s}^{\mu}$, we can perform the gaussian integrals on the internal and external weights:

$$\begin{aligned} Z = & \int \prod_{\mu}^{P} \frac{\mathrm{d}s^{\mu} \, \mathrm{d}\bar{s}^{\mu}}{2\pi} e^{-\frac{\beta}{2} \sum_{\mu} (y^{\mu} - s^{\mu})^2 + i \sum_{\mu}^{P} s^{\mu} \bar{s}^{\mu}} \\ & \times \left\{ \int \prod_{\mu}^{P} \frac{\mathrm{d}h^{\mu} \, \mathrm{d}\bar{h}^{\mu}}{2\pi} e^{i \sum_{\mu}^{P} h^{\mu} \bar{h}^{\mu} - \frac{1}{2\lambda_1 N_1} \left[\sum_{\mu}^{P} \bar{s}^{\mu} \sigma(h^{\mu}) \right]^2} e^{-\frac{1}{2\lambda_0 N_0} \sum_{i_0}^{N_0} \left(\sum_{\mu}^{P} \bar{h}^{\mu} x_{i_0}^{\mu} \right)^2} \right\}^{N_1} , \end{aligned} \tag{B.15}$$

where we used the fact that the integrals on $h_{i_1}^{\mu}$ and $\bar{h}_{i_1}^{\mu}$ can be factorized on the index i_1. The integral over the $\bar{h}^{\mu}$ is Gaussian and can be solved:

$$\int \prod_{\mu}^{P} \frac{\mathrm{d}\bar{h}^{\mu}}{2\pi} e^{i \sum_{\mu}^{P} h^{\mu} \bar{h}^{\mu} - \frac{1}{2\lambda_0 N_0} \sum_{i_0}^{N_0} \left(\sum_{\mu}^{P} \bar{h}^{\mu} x_{i_0}^{\mu} \right)^2} = P_1(\{h^{\mu}\}) , \tag{B.16}$$

where

$$P_1(\{h^{\mu}\}) = \frac{e^{-\frac{1}{2} \sum_{\mu,\nu}^{P} h^{\mu} \left[C^{-1} \right]_{\mu,\nu} h^{\mu}}}{\sqrt{(2\pi)^P \det C}} , \; C_{\mu\nu} = \frac{1}{\lambda_0 N_0} \sum_{i_0}^{N_0} x_{i_0}^{\mu} x_{i_0}^{\nu} . \tag{B.17}$$

This last step requires the covariance matrix C to be invertible. Note that this is false as soon

[1] One can map a system with non-zero biases in a zero-bias one increasing by one the dimensions of the input and of the activations at each layer. The original biases are then trivially mapped in the extra weights of the augmented system.

as $P > N_0$, but adding a small diagonal term to C solves the issue. One can explicitly check that the final result does not depend on this extra regularization.

To deal with the integral over h^μ we can include a further Dirac delta identity for the random variable $q = 1/\sqrt{\lambda_1 N_1} \sum_\mu \bar{s}^\mu \sigma\left(h^\mu\right)$. This leaves us with the problem of finding the probability density $P(q)$. In the limit defined in (4.1), this is exactly the same setting of the Breuer-Major theorems [99, 113, 114]. As such, it is sufficient that both the (regularized) covariance C and the activation function σ satisfy the hypotheses of the theorem to guarantee that the probability distribution $P(q)$ converges in distribution to a Gaussian:

$$\begin{aligned} P(q) &= \int \mathrm{d}^P h \, P_1(\{h^\mu\}) \delta\left[q - \frac{1}{\sqrt{\lambda_1 N_1}} \sum_\mu \bar{s}^\mu \sigma\left(h^\mu\right)\right], \\ P(q) &\to \mathcal{N}_q(0, Q)\,. \end{aligned} \tag{B.18}$$

with variance

$$\begin{aligned} Q(\bar{s}, C) &= \frac{1}{\lambda_1 N_1} \sum_{\mu,\nu}^{P} \bar{s}^\mu \left[\int \mathrm{d}^P h \, P_1(\{h^\rho\}) \sigma(h^\mu) \sigma(h^\nu)\right] \bar{s}^\nu \\ &= \frac{1}{\lambda_1 N_1} \sum_{\mu,\nu}^{P} \bar{s}^\mu K_{\mu\nu}(C) \bar{s}^\nu\,. \end{aligned} \tag{B.19}$$

One can show that there exist special configurations $\bar{s}$ in the domain of integration for which we are not allowed to invoke a Gaussian equivalence (see for instance the discussion at the end of Sec. C.1). In this derivation, we are assuming that the contribution of these special configurations to the effective action is negligible in the thermodynamic limit. Here we have also assumed that the variable q has zero mean, a condition verified as long as

$$\int \mathrm{d}^P h \, P_1(\{h^\mu\}) \sigma(h^\nu) = 0\,, \tag{B.20}$$

that is whenever σ is zero-mean; for a more general derivation, relevant for finite-mean activation functions such as ReLU, see the supplemental material C.3.

Now we can integrate over the variable q and obtain:

$$\left[\int \frac{\mathrm{d}q\, e^{-\frac{q^2}{2}-\frac{q^2}{2Q(\bar{s},C)}}}{\sqrt{2\pi Q(\bar{s},C)}}\right]^{\frac{N_1}{2}} = [Q(\bar{s},C)+1]^{-\frac{N_1}{2}} \ . \tag{B.21}$$

In the general case of finite $\alpha_1 = P/N_1$, we are only left with the integrals in s^μ and $\bar{s}^\mu$. To solve them it is convenient to introduce one final Dirac delta identity:

$$1 = \int dQ\, \delta\left[Q - \frac{1}{\lambda_1 N_1}\sum_{\mu,\nu}\bar{s}^\mu K(C)_{\mu\nu}\bar{s}^\nu\right], \tag{B.22}$$

where $Q \geq -1$ is now an integration variable and not a function of $\bar{s}$, so that we have removed the explicit dependence on $\sqrt{Q(\bar{s},C)+1}$ in the partition function. Finally, the integrals in s^μ and $\bar{s}^\mu$ are Gaussian once another integral representation of the delta via a conjugate variable $\bar{Q}$ is inserted. This allows us to get the final effective action obtained in equation (4.7). Notice that the effective action (4.7) has been firstly derived for deep linear networks in Ref. [9]. The non-asymptotic evaluation of the partition function in the linear case is given in Ref. [98] in terms of Meyer-G functions. An effective action for globally gated deep linear networks has been also derived [96].

B.2.1 Minimising the action

Equation (4.7) is solved using the saddle-point method, since $N_1 \to \infty$, which amounts to finding the solutions Q^*, $\bar{Q}^*$ of the system of equations $\partial_Q S = 0$, $\partial_{\bar{Q}} S = 0$. In the zero-temperature limit, we can find the analytical solution of the saddle-point equations. Using the fact that the kernel K has only positive eigenvalues (in the asymptotic regime α_1, α_0 finite), we get:

$$\bar{Q} = \frac{1}{1+Q}\,, \qquad Q = +\frac{\alpha_1}{\bar{Q}} - \frac{\alpha_1}{\bar{Q}^2}\frac{1}{P}y^T\left(\frac{K}{\lambda_1}\right)^{-1}y\,. \tag{B.23}$$

Given the condition $Q \geq -1$, the unique exact solution for $\bar{Q}$ is positive and reads:

$$\bar{Q}^* = \frac{\sqrt{(\alpha_1 - 1)^2 + 4\alpha_1 \frac{1}{P} y^T \left(\frac{K}{\lambda_1}\right)^{-1} y} - (\alpha_1 - 1)}{2} \,. \tag{B.24}$$

Consistently, the infinite-width limit is re-obtained for $\alpha_1 \to 0$ and corresponds to the particular solution $Q^* = 0$, $\bar{Q}^* = 1$.

B.2.2 Predictors statistics

The main observable we are interested in is the generalisation error (1.32). We can proceed along the same lines of the calculation performed in Sec. B.2 introducing, other than the variables s^μ, h_i^μ defined by (B.13), (B.14), additional variables s^0, h_i^0 that describe output and pre-activations of the new test example. We thus get:

$$\begin{aligned}
\langle \epsilon_{\rm g}(\mathbf{x}^0, y^0) \rangle = \frac{1}{Z} \int \frac{\mathrm{d}s^0 \, \mathrm{d}\bar{s}^0}{2\pi} \int \prod_{\mu=1}^{P} \frac{\mathrm{d}s^\mu \, \mathrm{d}\bar{s}^\mu}{2\pi} \, (y^0 - s^0)^2 \\
\times e^{-\frac{\beta}{2} \sum_{\mu=1}^{P} (y^\mu - s^\mu)^2 + i \sum_{\mu=1}^{P} s^\mu \bar{s}^\mu + i s^0 \bar{s}^0} \\
\times \left[1 + \frac{1}{\lambda_1 N_1} \left(\sum_{\mu,\nu=1}^{P} \bar{s}^\mu K_{\mu\nu} \bar{s}^\nu 2\bar{s}^0 \sum_{\mu=1}^{P} \bar{s}^\mu \kappa_\mu(\mathbf{x}^0) + (\bar{s}^0)^2 \kappa_0(\mathbf{x}^0) \right) \right]^{-\frac{N_1}{2}} ,
\end{aligned} \tag{B.25}$$

where κ_μ and κ_0 are respectively the train-test and the test-test kernel integrals defined as in (B.19) when the covariance matrix involves the test input, namely:

$$\kappa_\mu = \int \frac{dt_1 dt_2}{\sqrt{(2\pi)^2 \det \tilde{C}_\mu}} e^{-\frac{1}{2} \mathbf{t}^T \tilde{C}_\mu^{-1} \mathbf{t}} \sigma(t_1) \sigma(t_2) \,, \qquad \kappa_0 = \int \frac{dt}{\sqrt{2\pi C_{00}}} e^{-\frac{t^2}{2C_{00}}} \sigma(t)^2 \,, \tag{B.26}$$

where

$$\tilde{C}_\mu = \begin{pmatrix} C_{\mu\mu} & C_{\mu 0} \\ C_{\mu 0} & C_{00} \end{pmatrix}, \quad C_{\mu 0} = \frac{1}{\lambda_0 N_0} \sum_{i_0}^{N_0} x^\mu_{i_0} x^0_{i_0}, \tag{B.27}$$

$$C_{00} = \frac{1}{\lambda_0 N_0} \sum_{i_0}^{N_0} (x^0_{i_0})^2 . \tag{B.28}$$

Now we can introduce the order parameters Q and $\bar{Q}$ via equation (B.22) and their Fourier representation and perform the integration over all the s^μ, $\bar{s}^\mu$ and over the $\bar{s}^0$. Doing so yields a single integral in s^0 and integrals on Q and $\bar{Q}$.

$$\langle \epsilon_{\mathrm{g}}(\mathbf{x}^0, y^0) \rangle = \frac{1}{Z} \int \frac{\mathrm{d}Q\, \mathrm{d}\bar{Q}}{2\pi} e^{-\frac{N_1}{2} S(Q,\bar{Q})} \int \frac{\mathrm{d}s^0 (y^0 - s^0)^2}{\sqrt{2\pi\sigma^2}} e^{-\frac{(s^0+\Gamma_1)^2}{2\sigma_1^2}}, \tag{B.29}$$

with

$$\begin{aligned} \Gamma_1 &= \frac{\bar{Q}}{\lambda_1} \sum_{\mu\nu} \kappa_\mu(\mathbf{x}^0) \left(\frac{\mathbb{1}}{\beta} + \frac{\bar{Q}}{\lambda_1} K \right)^{-1}_{\mu\nu} y_\nu, \\ \sigma_1^2 &= \frac{\bar{Q}}{\lambda_1} \left[\kappa_0(\mathbf{x}^0) - \frac{\bar{Q}}{\lambda_1} \sum_{\mu\nu} \kappa_\mu(\mathbf{x}^0) \left(\frac{\mathbb{1}}{\beta} + \frac{\bar{Q}}{\lambda_1} K \right)^{-1}_{\mu\nu} \kappa_\nu(\mathbf{x}^0) \right] \end{aligned} \tag{B.30}$$

Taking the $\beta \to \infty$ limit in equations (B.30) yields the expressions in (4.18).

B.2.3 Generalisation to deep neural networks with a finite number of hidden layers $L > 1$ and zero-mean activation

In the same spirit of the 1HL calculation, we introduce L sets of auxiliary variables $h^\mu_{i_\ell}$ (where $i_\ell = 1, \ldots, N_\ell$) that are equal to the pre-activations at each layer. The strategy to perform the calculation is to show that the probability distribution of the preactivations at each layer $P_\ell(\{h^\mu_{i_\ell}\})$ can be computed recursively, starting from the input layer. We notice that this is conceptually different from the backpropagating kernel renormalisation group introduced in Ref. [9]. It is still a kernel renormalisation group, but forward-propagating, and represents

a generalisation to NNTPs of the kernel recurrence arising in NNGPs [27]. In practice, our approach amounts to a systematic, layer-by-layer description of the pre-activation statistics by the Student's t distribution that we have shown to appear in the 1HL case. This can be seen as a quantitative correction to the standard Gaussian statistics that is recovered in the infinite width limit. At the moment we are not able to re-derive the same result using the backpropagating method introduced in [9].

Let us start by integrating the weights of the first layer. This defines a probability distribution over the pre-activations of the first layer via:

$$P_1(\{h^\mu_{i_1}\}) = \int \mathcal{D}W^{(1)} \prod_{i_1,\mu} \delta\left(h^\mu_{i_1} - \frac{1}{\sqrt{N_0}} \sum_{i_0=1}^{N_0} W^{(1)}_{i_1 i_0} x^\mu_{i_0}\right) = \prod_{i_1=1}^{N_1} \frac{e^{-\frac{1}{2}\sum_{\mu\nu} h^\mu_{i_1} C^{-1}_{\mu\nu} h^\nu_{i_1}}}{\sqrt{(2\pi)^P \det C}}\,. \tag{B.31}$$

where C is defined in (B.17). This result is straightforward and it is valid for any N_0, P and N_1, since the prior for the weights is gaussian. At the second layer we have:

$$P_2(\{h^\mu_{i_2}\}) = \int \mathcal{D}W^{(2)} \mathcal{D}h_1 P_1(\{h^\mu_{i_1}\}) \prod_{i_2,\mu} \delta\left(h^\mu_{i_2} - \frac{1}{\sqrt{N_1}} \sum_{i_1=1}^{N_1} W^{(2)}_{i_2 i_1} \sigma(h^\mu_{i_1})\right) \tag{B.32}$$

We now introduce conjugate variables $\bar{h}^\mu_{i_2}$ to the activation of the second layer and the calculation proceeds as in the case of 1HL architectures. To make analytical progress we need to make two fundamental approximations: (i) assuming that the set of random variables $q_{i_2} = 1/(\sqrt{N_1}\lambda_1)\sum_\mu \bar{h}^\mu_{i_2}\sigma(h^\mu)$, where $\mathbf{h} \sim \mathcal{N}(0, C)$, is Gaussian-distributed; (ii) neglecting correlations between different pre-activations of the second hidden layer. In conclusion we get:

$$P_2(\{h^\mu_{i_2}\}) = \int \mathrm{d}Q_1\, \mathrm{d}\bar{Q}_1 e^{-\frac{N_1}{2}(-Q_1\bar{Q}_1 + \log(1+Q_1))} \prod_{i_2=1}^{N_2} \frac{e^{-\frac{1}{2}\sum_{\mu\nu} h^\mu_{i_2} \left(\bar{Q}_1 K(C)/\lambda_1\right)^{-1}_{\mu\nu} h^\nu_{i_2}}}{\sqrt{(2\pi)^P \det(\bar{Q}_1 K(C)/\lambda_1)}}\,, \tag{B.33}$$

where $K(C)$ is defined by equation (B.19). Notice that except for the integration over the two variables Q_1 and $\bar{Q}_1$, this is the same as the probability distribution of the 1HL system (B.17) if

we replace C with $\bar{Q}_1 K(C)/\lambda_1$. This reasoning can be easily iterated across layers and gives:

$$P_L(\{h^\mu_{i_L}\}) = \int \prod_{\ell=1}^{L-1} \mathrm{d}Q_\ell \, \mathrm{d}\bar{Q}_\ell e^{-\sum_{\ell=1}^{L-1} \frac{N_\ell}{2}\left[-Q_\ell \bar{Q}_\ell + \log(1+Q_\ell)\right]} \tag{B.34}$$

$$\times \prod_{i_L=1}^{N_L} \frac{e^{-\frac{1}{2}\sum_{\mu\nu} h^\mu_{i_L} \left(K^{(R)}_{L-1}(\{\bar{Q}_\ell\})\right)^{-1}_{\mu\nu} h^\nu_{i_L}}}{\sqrt{(2\pi)^P \det(K^{(R)}_{L-1}(\{\bar{Q}_\ell\}))}}\,, \tag{B.35}$$

where $K^{(R)}_\ell(\{\bar{Q}_\ell\})$ is a renormalised kernel that satisfies the recurrence relation in equation (4.44).

The computation of the generalisation error over a new example $(\mathbf{x}^0, y^0)$ gives:

$$\langle \epsilon_{\mathrm{g}}(\mathbf{x}^0, y^0)\rangle = (y^0 - \Gamma_L)^2 + \sigma_L^2 \tag{B.36}$$

where Γ_L and σ_L^2 are defined respectively in Eqs. (4.46) and (4.47). Note that $\kappa^{(R)}_{L\mu}$, $\kappa^{(R)}_{L0}$ are recursive kernels that generalise the train-test and test-test kernels. They are defined starting from equation (4.44) where the kernel K is now evaluated with the covariance matrix C involving train-test or test-test points. Note that L-hidden layers generalisation error is found replacing the 1HL kernel with its recursive generalisation (4.44).

APPENDIX C

Additional results for fully-connected networks in the proportional regime

C.1 1HL effective action and single output: special cases

In this section we report cases of activation functions for which we are able to evaluate analytically the probability distribution of the variable q, defined in the main text as

$$q = \frac{1}{\sqrt{\lambda N_1}} \sum_{\mu=1}^{P} \bar{s}^{\mu} \sigma(h^{\mu}) \,, \tag{C.1}$$

at fixed instance of the vector $\bar{s}$, clarifying the conditions to impose on the data for q to be Gaussian. Its characteristic function is defined as

$$\psi(t) = \mathbb{E}_q \left\{ \exp(iqt) \right\} = \mathbb{E}_h \left\{ \exp\left[\frac{it}{\sqrt{\lambda N_1}} \sum_{\mu} \bar{s}^{\mu} \sigma(h^{\mu}) \right] \right\} . \tag{C.2}$$

If q is Gaussian, then $\psi = \phi$, where

$$\phi(t) = \exp\left(-\frac{t^2 Q}{2}\right) = \exp\left(-\frac{t^2}{2\lambda N_1}\sum_{\mu,\nu}\bar{s}^\mu K_{\mu\nu}\bar{s}^\nu\right) \tag{C.3}$$

is the characteristic function of a Gaussian variable with variance given by

$$Q = \frac{1}{\lambda N_1}\sum_{\mu,\nu}\bar{s}^\mu K_{\mu\nu}\bar{s}^\nu\,, \qquad K_{\mu\nu} = \mathbb{E}_h[\sigma(h^\mu)\sigma(h^\nu)]\,. \tag{C.4}$$

C.1.1 Linear activation function: q is Gaussian at finite P, N_0, N_1

The case of $\sigma = \mathrm{id}$ has been already worked out in the literature, see [9, 98]. We report it here for reference, and to stress that our theory reduces to known cases as it should. Indeed, when the activation function is linear the average over h in Eq. (C.2) can be computed exactly at finite P, N_1, and gives

$$\psi_{\text{lin}}(t) = \exp\left(-\frac{t^2}{2\lambda N_1}\sum_{\mu,\nu}\bar{s}^\mu C_{\mu\nu}\bar{s}^\nu\right)\,. \tag{C.5}$$

Note that C is the value of the kernel for $\sigma = \mathrm{id}$. This is strictly true as long as C has no zero eigenvalue, so at least for $N_0 > P$; however, a small regularization proportional to the identity matrix can be added to C to avoid this problem.

This result is simply due to the fact that the sum of jointly Gaussian variables is Gaussian, which is true for generic Gram matrices C and any value of P, N_1, even far from the asymptotic limit $P \sim N_1$ large. In order to evaluate the remaining integrals over the order parameters at the saddle-point of the effective action, this limit is still required, as performed indeed in [9], to which our theory reduces; otherwise, for P, N_1 finite one can express the partition function exactly in terms of Meijer G-functions, see [98].

C.1.2 Quadratic activation function

Let us take now $C_{\mu\mu} = 1$ (normalized data) and quadratic (zero-mean) activation function:

$$\sigma(x) = x + a(x^2 - 1)\,. \tag{C.6}$$

The kernel is given by

$$K_{\mu\nu} = \mathbb{E}_h[\sigma(h^\mu)\sigma(h^\nu)] = C_{\mu\nu} + 2a^2(C_{\mu\nu})^2\,. \tag{C.7}$$

Also in this case the characteristic function in (C.2) can be evaluated exactly:

$$\psi_{\text{quad}}(t) = \frac{\exp\left\{-\frac{t^2}{2\lambda N_1}\bar{s}^\top C\left[\mathbb{1}_P - \frac{2iat}{\sqrt{\lambda N_1}}\operatorname{diag}(\bar{s})C\right]^{-1}\bar{s}\right\}}{\det[(\mathbb{1}_P - \frac{2iat}{\sqrt{\lambda N_1}}\operatorname{diag}(\bar{s})C)]^{1/2}}\exp\left(-\frac{iat}{\sqrt{\lambda N_1}}\sum_\mu \bar{s}^\mu\right)\,. \tag{C.8}$$

We can express the non-trivial matrices appearing in this expression as Neumann series:

$$\left[\mathbb{1}_P - \frac{2iat}{\sqrt{\lambda N_1}}\operatorname{diag}(\bar{s})C\right]^{-1} = \sum_{n=0}^{+\infty}\left(\frac{2iat}{\sqrt{\lambda N_1}}\right)^n [\operatorname{diag}(\bar{s})C]^n\,, \tag{C.9}$$

$$-\frac{1}{2}\operatorname{Tr}\log\left[\mathbb{1}_P - \frac{2iat}{\sqrt{\lambda N_1}}\operatorname{diag}(\bar{s})C\right] = -\sum_{n=1}^{+\infty}\frac{1}{n}\left(\frac{2ia}{\sqrt{\lambda N_1}}\right)^n \operatorname{Tr}\{[\operatorname{diag}(\bar{s})C]^n\}\,. \tag{C.10}$$

To prove Gaussianity, we need to require the following asymptotic behaviors:

$$\frac{1}{N_1^{1+n/2}}\bar{s}^\top C[\operatorname{diag}(\bar{s})C]^n\bar{s} = O(P/N_1^{1+n/2})\,, \tag{C.11}$$

$$\frac{1}{N_1^{n/2}}\operatorname{Tr}\{[\operatorname{diag}(\bar{s})C]^n\} = O(P/N_1^{n/2})\,, \tag{C.12}$$

so that in the regime where $\alpha_1 = P/N_1$ is finite only the $n = 0$ term counts for (C.9) and the $n = 1, 2$ terms for (C.10). Using

$$-\frac{1}{2}\operatorname{Tr}\log\left[\mathbb{1}_P - \frac{2iat}{\sqrt{\lambda N_1}}\operatorname{diag}(\bar{s})C\right] \approx \frac{iat}{\sqrt{\lambda N_1}}\sum_\mu \bar{s}^\mu - \frac{a^2t^2}{\lambda N_1}\sum_{\mu,\nu}\bar{s}_\mu(C_{\mu\nu})^2\bar{s}_\nu\,, \tag{C.13}$$

we get

$$\psi_{\text{quad}}(t) \sim \exp\left[-\frac{t^2}{2\lambda N_1}\sum_{\mu,\nu}\bar{s}^\mu K_{\mu\nu}\bar{s}^\nu\right] . \tag{C.14}$$

The conditions (C.11), (C.12) should be interpreted as hypothesis on the Gram matrix of the data C and on the realization of the vector $\bar{s}$ in order for the property of Gaussianity to hold. Let us see the simplest case of i.i.d. standard normal input data and $\bar{s}^\top = (1, \cdots, 1)$. Then, C is a Wishart matrix with a finite spectrum in the regime $P \sim N_0$ [115], and

$$\frac{1}{P}\operatorname{Tr}(C^n) = O(1)\,, \qquad \frac{1}{P}\sum_{\mu,\nu}(C^n)_{\mu\nu} = O(1)\,. \tag{C.15}$$

The first behaviour follows from the fact that the eigenvalues are $O(1)$, while the second can be proven using

$$C = O(1)\mathbb{1}_P + O(1/\sqrt{N_0})H\,, \tag{C.16}$$

where H is a symmetric random matrix with elements ± 1, or, more formally, exploiting the fact that the eigenvectors of a Wishart matrix are random and uniformly distributed on the sphere [116], so that

$$\frac{1}{P}\sum_{\mu,\nu}(C^n)_{\mu\nu} = \frac{1}{P}\sum_{\rho}\lambda_\rho^n\sum_{\mu}U_{\mu\rho}\sum_{\nu}U^{-1}_{\rho\nu} = O(1)\,, \tag{C.17}$$

where λ_ρ is the ρ-th eigenvalue of C and U the matrix whose ρ-th column is the corresponding eigenvector. Given that, properties (C.11), (C.12) follow and q is Gaussian.

In principle, Gaussianity can be also proven via diagrammatic techniques. Take for example

the quartic moment of the variable q in (C.1). One can see, via Wick's theorem, that

$$\begin{aligned}
&\mathbb{E}_h[\sigma(h^{\mu_1})\sigma(h^{\mu_2})\sigma(h^{\mu_3})\sigma(h^{\mu_4})] - (K_{\mu_1\mu_2}K_{\mu_3\mu_4} + K_{\mu_1\mu_3}K_{\mu_2\mu_4} + K_{\mu_1\mu_4}K_{\mu_2\mu_3}) = \\
&\quad 16a^4(C_{\mu_1\mu_2}C_{\mu_1\mu_3}C_{\mu_2\mu_4}C_{\mu_3\mu_4} + C_{\mu_1\mu_2}C_{\mu_1\mu_4}C_{\mu_2\mu_3}C_{\mu_3\mu_4} + C_{\mu_1\mu_3}C_{\mu_1\mu_4}C_{\mu_2\mu_3}C_{\mu_2\mu_4}) \\
&\quad + 4a^2(C_{\mu_1\mu_2}C_{\mu_1\mu_3}C_{\mu_2\mu_4} + C_{\mu_1\mu_2}C_{\mu_1\mu_3}C_{\mu_3\mu_4} + C_{\mu_1\mu_2}C_{\mu_1\mu_4}C_{\mu_2\mu_3} + \\
&\qquad C_{\mu_1\mu_2}C_{\mu_1\mu_4}C_{\mu_3\mu_4} + C_{\mu_1\mu_2}C_{\mu_2\mu_3}C_{\mu_3\mu_4} + C_{\mu_1\mu_2}C_{\mu_2\mu_4}C_{\mu_3\mu_4} + \\
&\qquad C_{\mu_1\mu_3}C_{\mu_1\mu_4}C_{\mu_2\mu_3} + C_{\mu_1\mu_3}C_{\mu_1\mu_4}C_{\mu_2\mu_4} + C_{\mu_1\mu_3}C_{\mu_2\mu_3}C_{\mu_2\mu_4} + \\
&\qquad C_{\mu_1\mu_3}C_{\mu_2\mu_4}C_{\mu_3\mu_4} + C_{\mu_1\mu_4}C_{\mu_2\mu_3}C_{\mu_2\mu_4} + C_{\mu_1\mu_4}C_{\mu_2\mu_3}C_{\mu_3\mu_4})\,,
\end{aligned} \tag{C.18}$$

while the quartic term from (C.3) involves only the diagrams

$$\begin{aligned}
&K_{\mu_1\mu_2}K_{\mu_3\mu_4} + K_{\mu_1\mu_3}K_{\mu_2\mu_4} + K_{\mu_1\mu_4}K_{\mu_2\mu_3} = \\
&\qquad 4a^4(C^2_{\mu_1\mu_2}C^2_{\mu_3\mu_4} + C^2_{\mu_1\mu_3}C^2_{\mu_2\mu_4} + C^2_{\mu_1\mu_4}C^2_{\mu_2\mu_3}) \\
&\qquad + 2a^2(C^2_{\mu_1\mu_2}C_{\mu_3\mu_4} + C_{\mu_1\mu_2}C^2_{\mu_3\mu_4} + C^2_{\mu_1\mu_3}C_{\mu_2\mu_4} + C_{\mu_1\mu_3}C^2_{\mu_2\mu_4} + C^2_{\mu_1\mu_4}C_{\mu_2\mu_3} + C_{\mu_1\mu_4}C^2_{\mu_2\mu_3}) \\
&\qquad + C_{\mu_1\mu_2}C_{\mu_3\mu_4} + C_{\mu_1\mu_3}C_{\mu_2\mu_4} + C_{\mu_1\mu_4}C_{\mu_2\mu_3}\,.
\end{aligned} \tag{C.19}$$

This is not surprising: the variables $\sigma(h^\mu)$ are not Gaussian due to the non-linearity. However, when summed over all the indices, the diagrams in Eq. (C.18) are of the form $\mathrm{Tr}C^4$ or $\sum_{\mu,\nu}(C^3)_{\mu\nu}$, both $O(P)$ under the hypothesis stated above, while the diagrams in (C.19) are of the form $(\sum_{\mu,\nu}(C^2)_{\mu\nu})^2$, $(\sum_{\mu,\nu}C_{\mu\nu})^2$ or $(\sum_{\mu,\nu}(C^2)_{\mu\nu})(\sum_{\mu,\nu}C_{\mu\nu})$, which are all $O(P^2)$ and leading over the first ones.

As long as $\bar{s}^\mu \sim O(1)$ for all μ, we do not expect the previous derivation to change. On the other hand, we point out that there exist special configurations $\bar{s}$, such as $\bar{s}^\top = (1, 0, \cdots, 0)$, for which this reasoning breaks down. As such, we are assuming that the contribution of these special configurations to the effective action is negligible in the thermodynamic limit.

C.2 Derivation of an effective action for 1HL neural networks with multiple outputs

In this section, we sketch the calculation to derive an effective action for 1HL neural networks with multiple outputs $\kappa > 1$. We stress that κ is finite and the case where the number of outputs scales with the width of the hidden layer N_1 has been the subject of investigations in [85, 86, 105]. We consider the following loss function:

$$\mathcal{L} = \frac{1}{2\kappa} \sum_{\mu=1}^{P} \sum_{a=1}^{\kappa} \left[y_a^\mu - (f_{\text{DNN}}(\mathbf{x}^\mu))_a \right]^2 + \mathcal{L}_{\text{reg}} \,, \tag{C.20}$$

$$\mathcal{L}_{\text{reg}} = \frac{\lambda_1}{2\beta} \|v\|^2 + \frac{\lambda_0}{2\beta} \|W\|^2 \,. \tag{C.21}$$

The partition function is defined as:

$$Z = \int \prod_{a,i_1}^{\kappa,N_1} \mathrm{d}v_{a,i_1} \prod_{i_1,i_0}^{N_1,N_0} \mathrm{d}w_{i_1 i_0} \, \exp\Bigg\{ -\frac{\lambda_1}{2} \|v\|^2 - \frac{\lambda_0}{2} \|w\|^2 \tag{C.22}$$

$$- \frac{\beta}{2\kappa} \sum_{\mu}^{P} \sum_{a}^{\kappa} \left[y_a^\mu - \frac{1}{\sqrt{N_1}} \sum_{i_1}^{N_1} v_{a,i_1} \sigma\left(\sum_{i_0}^{N_0} \frac{w_{i_1,i_0} x_{i_0}^\mu}{\sqrt{N_0}} \right) \right]^2 \Bigg\} \,. \tag{C.23}$$

We can decouple the layers in the loss through the addition of Dirac deltas, noticing that there will be one additional index a for the outputs.

$$\begin{aligned} Z = & \int \prod_{\mu,a}^{P,\kappa} \frac{\mathrm{d}s_a^\mu \, \mathrm{d}\bar{s}_a^\mu}{(2\pi)} \, \exp\left\{ -\frac{\beta}{2\kappa} \sum_{\mu,a}^{P,\kappa} (y_a^\mu - s_a^\mu)^2 + i \sum_{\mu,a}^{P,\kappa} s_a^\mu \bar{s}_a^\mu \right\} \int \prod_{\mu,i_1}^{P,N_1} \frac{\mathrm{d}h_{i_1}^\mu \, \mathrm{d}\bar{h}_{i_1}^\mu}{(2\pi)} \exp\left\{ i \sum_{\mu,i_1}^{P,N_1} h_{i_i}^\mu \bar{h}_{i_1}^\mu \right\} \\ & \int \prod_{a,i_1}^{\kappa,N_1} \mathrm{d}v_{a,i_1} \exp\left\{ -\frac{\lambda_1}{2} \|v\|^2 - i \sum_{a,\mu} \bar{s}_a^\mu \sum_{i_1} \frac{v_{a,i_1} h_{i_1}^\mu}{\sqrt{N_1}} \right\} \\ & \int \prod_{i_1,i_0}^{N_1,N_0} \mathrm{d}w_{i_1,i_0} \exp\left\{ -\frac{\lambda_0}{2} \|w\|^2 - i \sum_{i_1,\mu} \bar{h}_{i_1}^\mu \sum_{i_0} \frac{w_{i_1,i_0} x_{i_0}^\mu}{\sqrt{N_1}} \right\} \,. \end{aligned} \tag{C.24}$$

The integrals over the weights w_{i_1,i_0} and v_{a,i_1} are Gaussian and can be performed. As in the single-output case we can factorize the integrals in $h^\mu_{i_1}$ and $\bar{h}^\mu_{i_1}$ over the index i_1:

$$Z = \int \prod_{\mu,a} \frac{\mathrm{d}s^\mu_a \, \mathrm{d}\bar{s}^\mu_a}{2\pi} e^{-\frac{\beta}{2\kappa}\sum_{\mu,a}\left(y^\mu_a - s^\mu_a\right)^2 + i\sum_{\mu,a} s^\mu_a \bar{s}^\mu_a} \left\{ \int \prod_{\mu}^{P} \frac{\mathrm{d}h^\mu \, \mathrm{d}\bar{h}^\mu}{2\pi} e^{i\sum_\mu^P h^\mu \bar{h}^\mu - \frac{1}{2\lambda_1 N_1}\sum_a^\kappa \left(\sum_\mu \bar{s}^\mu_a \sigma(h^\mu)\right)^2 - \frac{1}{2\lambda_0 N_0}\sum_{i_0}^{N_0}\left(\sum_\mu^P \bar{h}^\mu x^\mu_{i_0}\right)^2} \right\}^{N_1} . \tag{C.25}$$

Once the integrals over the variables $\bar{h}^\mu$ are performed we obtain that the h^μ are Gaussian-distributed with zero mean and covariance matrix C, in analogy with the single-output case. The critical step is to consider the joint probability distribution of the following random variables:

$$q_a = \frac{1}{\sqrt{\lambda_1 N_1}} \sum_\mu^P \bar{s}^\mu_a \sigma(h^\mu). \tag{C.26}$$

As in the single-output case, in the asymptotic proportional limit $P/N_1 \sim O(1)$ we can conjecture a Gaussian equivalence, based on the reasonable assumption that the BM theorem can be generalised to the multivariate case. We therefore have that $P(\{q_a\}) \to \mathcal{N}(0, Q)$ where now Q is the covariance matrix given by:

$$Q(\bar{s}, C)_{a,b} = \frac{1}{\lambda_1 N_1} \sum_{\mu,\nu}^P \bar{s}^\mu_a \left[\int \mathrm{d}^P h \, P_1(\{h^\rho\}) \sigma(h^\mu)\sigma(h^\nu) \right] \bar{s}^\nu_b = \frac{1}{\lambda_1 N_1} \sum_{\mu,\nu}^P \bar{s}^\mu_a K_{\mu\nu}(C) \bar{s}^\nu_b \tag{C.27}$$

and K/λ_1 is the NNGP kernel, as in the single-output case. We now integrate over the set of variables $\{q_a\}$:

$$\int \prod_a^\kappa \mathrm{d}q_a \frac{1}{\sqrt{\det(Q)}} e^{-\frac{1}{2}\sum_a^\kappa (q_a)^2 - \frac{1}{2}\sum_{a,b}^\kappa q_a Q^{-1}_{a,b} q_b} = \det\left(\mathbb{1}_\kappa + \mathbf{Q}\right)^{-\frac{1}{2}} . \tag{C.28}$$

Differently from the single-output case, we need to introduce a $\kappa \times \kappa$ matrix order parameter

$Q_{a,b}$ as:

$$1 = \int \prod_{a,b} \mathrm{d}Q_{a,b}\, \delta \left[Q_{a,b} - \frac{1}{\lambda_1 N_1} \sum_{\mu,\nu}^{P} \bar{s}_a^\mu K_{\mu\nu}(C) \bar{s}_b^\nu \right] \tag{C.29}$$

and its dual $\bar{Q}_{a,b}$ via the Fourier representation of the deltas:

$$Z = \int \mathrm{d}\mathbf{Q}\bar{\mathbf{Q}}\ \det\left[\mathbb{1}_\kappa + \mathbf{Q}\right]^{-\frac{N_1}{2}} e^{i\sum_{a,b} Q_{a,b}\bar{Q}_{a,b}} \int \prod_{a,\mu} \mathrm{d}\bar{s}_a^\mu \exp\left\{ \frac{i}{\lambda_1 N_1} \sum_{a,b} \bar{Q}_{a,b} \sum_{\mu,\nu} \bar{s}_a^\mu K_{\mu\nu} \bar{s}_b^\nu \right\}$$
$$\int \prod_{a,\mu} \mathrm{d}s_a^\mu e^{-\frac{\beta}{2\kappa}\sum_{a,\mu}\left(y_a^\mu - s_a^\mu\right)^2 + i\sum_{a,\mu} s_{a,\mu}\bar{s}_{a,\mu}}. \tag{C.30}$$

In conclusion, the integrals in s and $\bar{s}$ can be solved and we land with the following effective action S:

$$S(\mathbf{Q}, \bar{\mathbf{Q}}) = -\operatorname{Tr}[\mathbf{Q}\bar{\mathbf{Q}}^\top] + \operatorname{Tr}\log(\mathbb{1}_\kappa + \mathbf{Q}) + \frac{\alpha_1}{P} \operatorname{Tr}\log\frac{\beta}{\kappa}\left[\frac{\kappa}{\beta}\,\mathbb{1}_\kappa \otimes \mathbb{1}_P \right. \tag{C.31}$$
$$\left. + \frac{\bar{\mathbf{Q}} \otimes K}{\lambda_1}\right] + \frac{\alpha_1}{P} y^\top \left[\frac{\kappa}{\beta}\,\mathbb{1}_\kappa \otimes \mathbb{1}_P + \frac{\bar{\mathbf{Q}} \otimes K}{\lambda_1}\right]^{-1} y. \tag{C.32}$$

C.3 Generalising the effective action to finite-mean activation functions

In this section we show how the theory can be generalized in the case of finite-mean activation functions. In fact, up to this point, our derivation assumed that the integral of the activation function over a centered Gaussian is zero, i.e. the activation function is zero-mean. The goal of this section is to show that removing such hypothesis modifies the effective action in the asymptotic limit. Since ReLU activation belongs to this more general case, the findings of this section imply that Li-Sompolinsky heuristic theory [9] should be modified as well. As for the rest of the manuscript, we start by considering one hidden layer architectures and we later extend the result to L hidden layers.

The crucial difference wrt to the case studied in section 4.1 is that if the activation function is not zero-mean, also the random variable

$$q = \frac{1}{\sqrt{N_1 \lambda_1}} \sum_{\mu=1}^{P} \bar{s}^\mu \sigma(h^\mu) \tag{C.33}$$

has now a finite mean. In particular:

$$\langle q \rangle_{P(q)} = \frac{1}{\sqrt{N_1 \lambda_1}} \sum_{\mu=1}^{P} \bar{s}^\mu m^\mu \,, \qquad m^\mu = \int \frac{dt}{\sqrt{2\pi C_{\mu\mu}}} e^{-t^2/(2C_{\mu\mu})} \sigma(t) \,. \tag{C.34}$$

A straightforward calculation shows that the result for finite-mean activation is found by performing the replacement:

$$\frac{\bar{Q}}{\lambda_1} K \to K^{(R)}(Q, \bar{Q}) = \frac{\bar{Q}}{\lambda_1} K - \frac{\left(\bar{Q} - \frac{1}{1+Q}\right)}{\lambda_1} K^{(1)} \,, \qquad K^{(1)}_{\mu\nu} = m^\mu m^\nu \tag{C.35}$$

in the effective action in Eq. (4.7) of the main text. As such, the one-hidden layer action for finite-mean activation functions reads:

$$S_{\text{1HL}} = -Q\bar{Q} + \log(1+Q) + \frac{\alpha_1}{P} \operatorname{Tr} \log \beta \left[\frac{\mathbb{1}}{\beta} + K^{(R)}(Q, \bar{Q}) \right] + \frac{\alpha_1}{P} y^\top \left[\frac{\mathbb{1}}{\beta} + K^{(R)}(Q, \bar{Q}) \right]^{-1} y \tag{C.36}$$

It is worth noticing that while in the zero mean case there was a simple relations between Q and $\bar{Q}$ at any temperature, we now lose that property and the saddle-point equations are not exactly solvable anymore, not even in the zero temperature limit. On the contrary, one can check that in the infinite-width limit we recover the previous result $\bar{Q} = 1$, $Q = 0$ and the rank one matrix $K^{(1)}$ does not contribute to the generalization error, since it does always appear in combination with the scalar $\bar{Q} - 1/(1+Q)$ that vanishes in the infinite-width limit.

Let us move to the derivation of an effective action for L hidden layers. As for the derivation with zero-mean activation function, the key step is to understand how the joint probability of the pre-activations at layer ℓ is linked to the one at layer $\ell - 1$. While in the zero-mean

activation case, the key observation was that P_2 is related to P_1 by the replacement $C \to \bar{Q}_1 K(C)/\lambda_1$ (see Eq. B.33), here we find that the correct replacement is $C \to \bar{Q}_1 K(C)/\lambda_1 - (\bar{Q}_1 - 1/(1+Q_1))K^{(1)}/\lambda_1$. Differently from the zero-mean activation case, where the kernel at layer L was only depending on the variables $\{\bar{Q}_\ell\}$, here we find that the recurrence is given in terms of the $\{Q_\ell\}$ as well. In conclusion, this produces a more unpleasant action where all the $\{Q_\ell, \bar{Q}_\ell\}$ are coupled via the nested non-linear expression of the kernel. The explicit recurrence relation for finite-mean activation functions is given by:

$$K_\ell^{(R)} = \frac{\bar{Q}_\ell}{\lambda_\ell} K \circ \left[K_{\ell-1}^{(R)}\right] - \frac{\left(\bar{Q}_\ell - \frac{1}{1+Q_\ell}\right)}{\lambda_\ell} K^{(1)} \circ \left[K_{\ell-1}^{(R)}\right], \qquad K_0^{(R)} = C \tag{C.37}$$

and the effective saddle-point action reads:

$$\begin{aligned} S_{\text{DNN}} = &\sum_{\ell=1}^{L} \frac{\alpha_L}{\alpha_\ell} \left[-Q_\ell \bar{Q}_\ell + \log(1+Q_\ell)\right] + \frac{\alpha_L}{P} \text{Tr} \log \beta \left(\frac{\mathbb{1}}{\beta} + K_L^{(R)}(\{\bar{Q}_\ell, Q_\ell\})\right) \\ &+ \frac{\alpha_L}{P} y^T \left(\frac{\mathbb{1}}{\beta} + K_L^{(R)}(\{\bar{Q}_\ell, Q_\ell\})\right)^{-1} y \,. \end{aligned} \tag{C.38}$$

In view of the above considerations, it should be now clear that the heuristic Li-Sompolinsky theory (re-derived in the previous section) amounts to disregard all the additional terms $K^{(1)}$ that arise from the approach presented in this section.

APPENDIX D

Computations with the replicated ensemble

D.1 Computation of the partition function

We are interested in the computation of the partition function:

$$
\begin{aligned}
Z &= \int \prod_{\alpha}^{Y} d\theta_\alpha e^{-\beta \mathcal{L}(\{\theta_\alpha\})} \\
&= \int \prod_{\alpha\, i_0\, i_1} dw^{\alpha}_{i_1 i_0} \prod_{\alpha\, i_1} dv^{\alpha}_{i_1} \exp\left(-\frac{\beta}{2} \sum_{\alpha\mu} \left(y^\mu - f_{\theta_\alpha}(x^\mu)\right)^2 + \right. \\
&\qquad \left. - \frac{\lambda_1}{2} \sum_{\alpha,\beta,i_1} v^{\alpha}_{i_1} (M_1^{-1})_{\alpha\beta} v^{\beta}_{i_1} - \frac{\lambda_0}{2} \sum_{\alpha\beta, i_0 i_1} w^{\alpha}_{i_1 i_0} (M_0^{-1})_{\alpha\beta} w^{\beta}_{i_1 i_0} \right)
\end{aligned}
$$

where M_1 and M_0 can be any matrices for the time being, and $f_{\theta_\alpha}(x^\mu)$ is one of the Y identical 1HL networks defined in (4.23). The physical Bayesian interpretation is retrieved choosing M_1 and M_2 as in Eq. (4.26). I can now introduce the usual variables $h^{\mu\, a}_{i_1}$, $s^{\mu\, a}$, that now have an

extra replica index, as constraints through a collection of delta functions:

$$\prod_{i_1\,\mu\,\alpha}\delta\left(h_{i_1}^{\mu\,\alpha}-\frac{1}{\sqrt{N_0}}\sum_{i_0=1}^{N_0}w_{i_1 i_0}^{\alpha}x_{i_0}^{\mu}\right)\qquad\prod_{\mu\,\alpha}\delta\left(s^{\mu\,\alpha}-\frac{1}{\sqrt{N_1}}\sum_{i_1=1}^{N_1}v_{i_1}^{\alpha}\sigma\left(h_{i_1}^{\mu\,\alpha}\right)\right)\tag{D.1}$$

I can insert these constraints in the partition function directly in their exponential form, introducing also a set of conjugate variables $\bar{h}_{i_1}^{\mu\,a}$, $\bar{s}^{\mu\,a}$. After performing the gaussian integrations over the weights one gets:

$$\begin{aligned}Z=\mathcal{C}\int\prod_{\alpha\mu}d^2s^{\mu\alpha}e^{-\frac{\beta}{2}\sum_{\alpha\mu}(y^{\mu}-s^{\mu\alpha})^2+i\sum_{\alpha\mu}s^{\mu\alpha}\bar{s}^{\mu\alpha}}\\\times\left[\int\prod_{\alpha\mu}d^2h^{\mu\alpha}e^{+i\sum_{\alpha\mu}h^{\mu\,a}\bar{h}^{\mu\,a}}\right.\\\left.\times\;e^{-\frac{1}{2}\sum_{\alpha\beta\mu\nu}\bar{h}^{\mu\alpha}M_0^{\alpha\beta}C^{\mu\nu}\bar{h}^{\nu\beta}-\frac{1}{2\lambda_1 N_1}\sum_{\alpha\beta\mu\nu}\bar{s}^{\alpha\mu}\sigma(h^{\alpha\mu})M_1^{\alpha\beta}\bar{s}^{\beta\nu}\sigma(h^{\beta\nu})}\right]^{N_1}\end{aligned}\tag{D.2}$$

where $\mathcal{C}$ is a negligible constant, and I have also introduced the correlation matrix of the dataset, defined as:

$$C^{\mu\nu}=\frac{1}{\lambda_0 N_0}\sum_{i_0}x_{i_0}^{\mu}x_{i_0}^{\nu}\tag{D.3}$$

The integrals over $\bar{h}$ can be computed too:

$$\begin{aligned}Z=\mathcal{C}\int\prod_{\alpha\mu}d^2s^{\mu\alpha}e^{-\frac{\beta}{2}\sum_{\alpha\mu}(y^{\mu}-s^{\mu\alpha})^2+i\sum_{\alpha\mu}s^{\mu\alpha}\bar{s}^{\mu\alpha}}\\\times\left[\int\prod_{\alpha\,\mu}dh^{\mu\alpha}e^{-\frac{1}{2N_1\lambda_1}\sum_{\alpha\beta\mu\nu}\bar{s}^{\alpha\mu}\sigma(h^{\alpha\mu})M_1^{-1}\bar{s}^{\beta\nu}\sigma(h^{\beta\nu})}\right.\\\left.\times\;\frac{e^{-\frac{1}{2}\sum_{\alpha\beta}\mathbf{h}_{\alpha}^{T}\Gamma^{-1}\mathbf{h}_{\beta}}}{\sqrt{(2\pi)^{YP}(\mathrm{det}C)^{Y}(\mathrm{det}M_0)^{P}}}\right]^{N_1}\end{aligned}\tag{D.4}$$

Where $\Gamma = C \otimes M_0$ is a $YP \times YP$ matrix.The integral in square brackets:

$$
\begin{aligned}
[\dots]^{N_1} &= \left[\int \prod_{\alpha} dq^{\alpha}\, p(\{q\}) e^{-\frac{1}{2}\sum_{\alpha\beta} q^{\alpha} M_1^{\alpha\beta} q^{\beta}} \right]^{N_1} \\
p(\{q\}) &= \int \prod_{\alpha\,\mu} dh^{\mu\alpha} \frac{e^{-\frac{1}{2}\sum_{\alpha} \mathbf{h}_{\alpha}^{T} \Gamma^{-1} \mathbf{h}_{\beta}}}{\sqrt{(2\pi)^{YP} (\det \Gamma)^{Y}}} \delta\left(q^{\alpha} - \frac{1}{\sqrt{\lambda_1 N_1}} \sum_{\mu} \bar{s}^{\alpha\mu} \sigma(h^{\alpha\mu}) \right)
\end{aligned}
\tag{D.5}
$$

Relying on the Breuer-Major theorem [99], we conjecture that this distribution converges to a gaussian one for large N_1. Therefore we have:

$$
[\dots]^{N_1} = \left[\int \prod_{\alpha} dq^{\alpha} \frac{e^{-\frac{1}{2}\sum_{\alpha\beta} q^{\alpha} (M_1 + Q^{-1})_{\alpha\beta} q^{\beta}}}{\sqrt{(2\pi)^{Y} (\det Q)}} \right]^{N_1} = [\det(Q M_1 + \mathbb{I})]^{-\frac{N_1}{2}}
\tag{D.6}
$$

The $Y \times Y$ covariance matrix elements $Q_{\alpha\beta}$ read:

$$
Q_{\alpha\beta}(\bar{s}^{\alpha}, \bar{s}^{\beta}) = \langle q^{\alpha} q^{\beta} \rangle \equiv \frac{1}{\lambda_1 N_1} \sum_{\mu\nu} \bar{s}^{\alpha\mu} K_{\mu\nu}^{\alpha\beta} \bar{s}^{\beta\nu}
\tag{D.7}
$$

The partition function reads

$$
Z = \mathcal{C} \int \prod_{\alpha\mu} d^2 s^{\mu\alpha} e^{-\frac{\beta}{2}\sum_{\alpha\mu}(y^{\mu} - s^{\mu\alpha})^2 + i \sum_{\alpha\mu} s^{\mu\alpha} \bar{s}^{\mu\alpha}} [\det(\mathbb{I} + Q M_1)]^{-\frac{N_1}{2}}
\tag{D.8}
$$

To make progress in the calculation, one needs to introduce a set of deltas (and their conjugates) for the covariance matrix elements:

$$
\prod_{\alpha\beta} \delta\left(Q^{\alpha\beta} - \frac{1}{\lambda_1 N_1} \sum_{\mu\nu} \bar{s}^{\alpha\mu} K_{\mu\nu}^{\alpha\beta} \bar{s}^{\beta\nu} \right)
\tag{D.9}
$$

The kernel K inherits a replica symmetric structure:

$$
K = K^{1} \otimes \mathbb{I}_{Y} + K^{0} \otimes \mathbb{O}_{Y},
\tag{D.10}
$$

where K^1andK^2 are $P \times P$ matrices with entries:

$$K^1_{\mu\nu} = \int dp\left(\{\mathbf{h}\}\right)\ \sigma(h^{\mu\alpha})\sigma(h^{\nu\alpha}) \tag{D.11}$$

$$K^0_{\mu\nu} = \int dp\left(\{\mathbf{h}\}\right)\ \sigma(h^{\mu\alpha})\sigma(h^{\nu\beta}) \tag{D.12}$$

$$p\left(\{\mathbf{h}\}\right) = \frac{1}{\sqrt{(2\pi)^{P(Y-1)}\det\Gamma}}\, e^{-\frac{1}{2}\sum_{\alpha,\beta}^{Y\,P}{}_{\mu\nu} h^{\mu\alpha}(\Gamma^{-1})^{\alpha\beta}_{\mu\nu}h^{\mu\beta}}. \tag{D.13}$$

Similarly to the unreplicated case, it can be shown that:

$$K^1_{\mu\nu} = K^1_{\mu\nu}\left(\Gamma^1_{\mu\mu},\Gamma^1_{\mu\nu},\Gamma^1_{\nu\nu}\right) \qquad K^0_{\mu\nu} = K^1_{\mu\nu}\left(\Gamma^1_{\mu\mu},\Gamma^0_{\mu\nu},\Gamma^1_{\nu\nu}\right), \tag{D.14}$$

$$\Gamma^1 = \frac{\gamma_0+\lambda_0}{\lambda_0+Y\gamma_0}C \qquad \Gamma^0 = \frac{\gamma_0}{\lambda_0+Y\gamma_0}C. \tag{D.15}$$

This is explicitely shown in Appendix B (Eqs. B.11, B.10). The partition function reads:

$$\begin{aligned} Z \propto & \int \prod_{\alpha,\beta} dQ_{\alpha\beta} e^{\frac{N_1}{2}\mathrm{Tr}(Q\bar{Q})-\frac{N_1}{2}\log\det(I+QM_1)} \\ & \times \int \prod_{\alpha\mu} d^2 s^{\mu\alpha} e^{i\sum_{\alpha\mu} s^{\mu\alpha}\bar{s}^{\mu\alpha}-\frac{\beta}{2}\sum_{\alpha\mu}(y^\mu-s^{\mu\alpha})^2-\frac{1}{2\lambda_1}\sum_{\alpha\beta,\mu\nu}\bar{s}^{\mu\alpha}\bar{Q}_{\alpha\beta}K^{\alpha\beta}_{\mu\nu}\bar{s}^{\nu\beta}} \end{aligned} \tag{D.16}$$

Isolating the renormalized kernel:

$$\left[K^{(\mathrm{R})}(\bar{Q})\right]^{\alpha\beta}_{\mu\nu} = \bar{Q}_{\alpha\beta}/\lambda_1 K^{\alpha\beta}_{\mu\nu} \tag{D.17}$$

and performing the remaining integrations in s and $\bar{s}$ respectively, one obtains the result in equation 4.27.

D.1.1 Infinite-width limit saddle point equations

In the infinite-width limit where $N_1 \gg P$, we have that the terms of order $\alpha = P/N_1$do not count. The action simplifies:

$$A(Q,\bar{Q}) = -\mathrm{Tr}(Q\bar{Q}) + \log\det\left(I+QM_1\right) \tag{D.18}$$

The saddle point equations read:

$$\frac{\partial A}{\partial Q} = -\bar{Q} + (1 + QM_1)^{-1} M_1 \qquad \frac{\partial A}{\partial \bar{Q}} = Q \tag{D.19}$$

That are easily solved to yield:

$$Q^* = 0 \qquad \bar{Q}^* = M_1 \tag{D.20}$$

Note that an unreplicated system in the infinite-width limit would have $M_1 = \mathbb{1}$, therefore there would be no contribution from the off-diagonal terms of $\bar{Q}$.

D.2 Average accuracy

We now can compute the average accuracy the replica ensemble, on a new test example $\{\boldsymbol{x}^0, y^0\}$. The average we want to compute:

$$\begin{aligned}
\epsilon_{acc} &= \left\langle \theta\left(\frac{1}{Y}\sum_{\alpha} y^0 f_{\theta_\alpha}(x^0)\right)\right\rangle \\
&= \frac{1}{Z}\int \prod_{\alpha=1}^{Y} d\theta_\alpha \theta\left(\frac{1}{Y}\sum_{\alpha} y^0 f_{\theta_\alpha}(x^0)\right) e^{-\beta\mathcal{L}(\{\theta_\alpha\})} \\
&= \frac{1}{Z}\int \prod_{\alpha\, i_0\, i_1} dw^\alpha_{i_1 i_0} \prod_{\alpha\, i_1} dv^\alpha_{i_1}\; \theta\left(\sum_{\alpha} y^0 f_{\theta_\alpha}(x^0)\right) e^{-\frac{\beta}{2}\sum_{\alpha\mu}(y^\mu - f_{\theta_\alpha}(x^\mu))^2} \\
&\qquad \times e^{-\frac{\lambda_1}{2}\sum_{\alpha,\beta,i_1} v^\alpha_{i_1}(M_1^{-1})_{\alpha\beta} v^\beta_{i_1} - \frac{\lambda_0}{2}\sum_{\alpha\beta,i_0 i_1} w^\alpha_{i_1 i_0}(M_0^{-1})_{\alpha\beta} w^\beta_{i_1 i_0}}
\end{aligned} \tag{D.21}$$

We can now introduce some dirac deltas:

$$\prod_{i_1\,\mu\,\alpha} \delta\left(h^{\mu\,\alpha}_{i_1} - \frac{1}{\sqrt{N_0}}\sum_{i_0=1}^{N_0} w^\alpha_{i_1 i_0} x^\mu_{i_0}\right) \qquad \prod_{\mu\,\alpha} \delta\left(s^{\mu\,\alpha} - \frac{1}{\sqrt{N_1}}\sum_{i_1=1}^{N_1} v^\alpha_{i_1}\sigma\left(h^{\mu\,\alpha}_{i_1}\right)\right) \tag{D.22}$$

$$\prod_{i_1\alpha} \delta\left(h^{0\alpha}_{i_1} - \frac{1}{\sqrt{N_0}}\sum_{i_0=1}^{N_1} w^\alpha_{i_1 i_0} x^0_{i_0}\right) \qquad \prod_{\alpha} \delta\left(s^{0\alpha} - \frac{1}{\sqrt{N_1}}\sum_{i_1=1}^{N_1} v^\alpha_{i_1}\sigma\left(h^0_{i_1}\right)\right) \tag{D.23}$$

And compute the gaussian integrals over the weights. Note that, when it's not specified, $\sum_\mu$indicates that $\mu = 1 \dots P$. The average accuracy:

$$\begin{aligned}\epsilon_{acc} = \frac{1}{Z}\int \prod_{\alpha\mu=0} d^2 s^{\mu\alpha}\theta\left(\frac{1}{Y}\sum_\alpha y^0 f_{\theta_\alpha}(x^0)\right) e^{-\frac{\beta}{2}\sum_{\alpha\mu}(y^\mu - s^{\mu\alpha})^2 + i\sum_{\alpha\mu=0} s^{\mu\alpha}\bar{s}^{\mu\alpha}} \\ \times \left[\int \prod_{\alpha\mu=0} d^2 h^{\mu\alpha} e^{i\sum_{\alpha\mu=0} h^{\mu\alpha}\bar{h}^{\mu\alpha}}\right. \\ \left. \times\ e^{-\frac{1}{2}\sum_{\alpha\beta\mu\nu=0}\bar{h}^{\mu\alpha}M_0^{\alpha\beta}C^{\mu\nu}\bar{h}^{\nu\beta} - \frac{1}{2\lambda_1 N_1}\sum_{\alpha\beta\mu\nu=0}\bar{s}^{\alpha\mu}\sigma(h^{\alpha\mu})M_1^{\alpha\beta}\bar{s}^{\beta\nu}\sigma(h^{\beta\nu})}\right]^{N_1}\end{aligned} \tag{D.24}$$

Where now the covariance matrix is extended to the new example $\mathbb{x}^0$, becoming $P+1$-dimensional symmetrix matrix with the extra row (and column) given by the vector $(\mathbf{x}^\mu \cdot \mathbf{x}^0)/\lambda_0 N_0,\ \mathbf{x}^\mu \in X)$. After performing the integrals in $\bar{h}$and h, with the usual gaussian approximation justified by the BM theorem [99], the test accuracy function reads:

$$\begin{aligned}\epsilon_{acc} \propto \frac{1}{Z}\int \prod_{\alpha\mu=0} d^2 s^{\mu\alpha}\theta\left(\frac{1}{Y}\sum_\alpha y^0 f_{\theta_\alpha}(x^0)\right) \\ \times\ e^{-\frac{\beta}{2}\sum_{\alpha\mu}(y^\mu - s^{\mu\alpha})^2 + i\sum_{\alpha\mu=0} s^{\mu\alpha}\bar{s}^{\mu\alpha}} \left[\det(\mathbb{1} + QM_1)\right]^{-\frac{N_1}{2}}\end{aligned} \tag{D.25}$$

Where the Q is the covariance matrix of the distribution in D.5, its elements $Q_{\alpha\beta}$ read:

$$Q_{\alpha\beta}(\bar{s}^\alpha, \bar{s}^\beta) = \langle q^\alpha q^\beta\rangle \equiv \frac{1}{\lambda_1 N_1}\sum_{\mu\nu=0}\bar{s}^{\alpha\mu}K_{\mu\nu}^{\alpha\beta}\bar{s}^{\beta\nu} \tag{D.26}$$

Note that here we have one extra contribution to the sum, given by the new test example, that is however negligible in the thermodynamic limit $P \to \infty$. Now we need to introduce a set of

deltas (and their conjugates) for the covariance matrix elements:

$$\prod_{\alpha\beta}\delta\left(Q^{\alpha\beta}-\frac{1}{\lambda_1 N_1}\sum_{\mu\nu=0}\bar{s}^{\alpha\mu}K^{\alpha\beta}_{\mu\nu}\bar{s}^{\beta\nu}\right) \tag{D.27}$$

$$\begin{aligned}\epsilon_{acc}\propto&\frac{1}{Z}\int\prod_{\alpha,\beta}dQ_{\alpha\beta}\theta\left(\frac{1}{Y}\sum_{\alpha}y^0 f_{\theta_\alpha}(x^0)\right)e^{\frac{N_1}{2}\mathrm{Tr}(Q\bar{Q})-\frac{N_1}{2}\log\det(I+QM_1)}\\&\times\int\prod_{\alpha\mu=0}d^2s^{\mu\alpha}e^{i\sum_{\alpha\mu=0}s^{\mu\alpha}\bar{s}^{\mu\alpha}-\frac{\beta}{2}\sum_{\alpha\mu}(y^\mu-s^{\mu\alpha})^2-\frac{1}{2\lambda_1}\sum_{\alpha\beta,\mu\nu=0}\bar{s}^{\mu\alpha}\bar{Q}_{\alpha\beta}K^{\alpha\beta}_{\mu\nu}\bar{s}^{\nu\beta}}\end{aligned} \tag{D.28}$$

We want to separate the contribution from the trainset elements to that of x^0. Let me define $S\equiv K^{(\mathrm{R})}$ as in D.17, and $\tilde{S}=K(\mathbf{x}^\mu\mathbf{x}^0)\,\mu=0\dots P$ with the Kernel function in 1.42. We can rewrite:

$$\epsilon_{acc}=\frac{\mathcal{C}}{Z}\int\prod_{\alpha,\beta}dQ_{\alpha\beta}e^{\frac{N_1}{2}\mathrm{Tr}(Q\bar{Q})-\frac{N_1}{2}\log\det(I+QM_1)} \tag{D.29}$$

$$\times\int\prod_{\alpha}d^2s^{0\alpha}\theta\left(\frac{1}{Y}\sum_{\alpha}y^0 f_{\theta_\alpha}(x^0)\right)e^{i\sum_\alpha s^{\mu 0}\bar{s}^{\mu 0}-\frac{1}{2}\sum_{\alpha\beta}\bar{s}^{0\alpha}\tilde{S}^{\alpha\beta}_{00}\bar{s}^{0\beta}} \tag{D.30}$$

$$\times\int\prod_{\alpha\mu=1}d^2s^{\mu\alpha}e^{i\sum_{\alpha\mu}s^{\mu\alpha}\bar{s}^{\mu\alpha}-\frac{\beta}{2}\sum_{\alpha\mu}(y^\mu-s^{\mu\alpha})^2-\frac{1}{2}\sum_{\alpha\beta,\mu\nu}\bar{s}^{\mu\alpha}S^{\alpha\beta}_{\mu\nu}\bar{s}^{\nu\beta}-\sum_{\alpha\beta,\mu}\bar{s}^{0\alpha}S^{\alpha\beta}_{0\mu}\bar{s}^{\mu\beta}} \tag{D.31}$$

We now can perform respectively the integrations over $s^{\mu\alpha}$ with $\mu\neq 0$, and all the $\bar{s}^{\mu\alpha}$, to obtain:

$$\begin{aligned}\epsilon_{acc}=&\frac{\mathcal{C}}{Z}\int\prod_{\alpha,\beta}dQ_{\alpha\beta}e^{\frac{N_1}{2}\mathrm{Tr}(Q\bar{Q})-\frac{N_1}{2}\log\det[I+QM_1]-\frac{1}{2}\log\det\left[S+\frac{\mathbb{1}}{\beta}\right]-\sum_{\alpha\beta\mu\nu}y^\nu\left(\left(S+\frac{\mathbb{1}}{\beta}\right)^{-1}\right)^{\alpha\beta}_{\mu\nu}y^\mu}\\&\times\int\prod_{\alpha}ds^{0\alpha}\frac{\frac{1}{Y}\sum_\alpha y^0 f_{\theta_\alpha}(x^0)}{\sqrt{(2\pi)^Y\det\Sigma}}\\&\times e^{-\frac{1}{2}\sum_{\alpha\beta}s^{0\alpha}[\Sigma^{-1}]^{\alpha\beta}s^{0\beta}+\sum_\alpha s^{0\alpha}\sum_\beta m^\beta[\Sigma^{-1}]^{\alpha\beta}-\frac{1}{2}\sum_{\alpha\beta}m^\alpha[\Sigma^{-1}]^{\alpha\beta}m^\beta}\\=&\frac{\mathcal{C}}{Z}\int\prod_{\alpha,\beta}dQ_{\alpha\beta}e^{-\frac{N_1}{2}A(Q,\bar{Q})}H\left(\frac{-\mathrm{sign}(y^0)\sum_\alpha m^\alpha}{\sqrt{\sum_{\alpha\beta}\Sigma_{\alpha\beta}}}\right)\end{aligned}$$

and we have defined

$$\Sigma^{\alpha\beta} = \tilde{S}_{00}^{\alpha\beta} - \sum_{\mu\nu} \tilde{S}_{0\mu}^{\alpha\beta} \left[\left(S + \frac{\mathbb{1}}{\beta} \right)^{-1} \right]_{\mu\nu}^{\alpha\beta} \tilde{S}_{0\nu}^{\alpha\beta} \tag{D.32}$$

$$m^{\alpha} = \sum_{\alpha'\mu\nu} y^{\mu} \tilde{S}_{0\nu}^{\alpha'\alpha} \left[\left(S + \frac{\mathbb{1}}{\beta} \right)^{-1} \right]_{\mu\nu}^{\alpha'\alpha} \tag{D.33}$$

Note that the action $A(Q, \bar{Q})$ is the one defined in 4.27, therefore we have that the generalisation accuracy:

$$\epsilon_{acc} = H \left(\frac{-\text{sign}(y^0) \sum_{\alpha} m^{\alpha}}{\sqrt{\sum_{\alpha\beta} \Sigma_{\alpha\beta}}} \right) \tag{D.34}$$

That has to be evaluated at the saddle point $Q^*, \bar{Q}^*$as defined in D.20. In the zero temperature limit $\beta \to \infty$, the expressions in further simplify:

$$\Sigma^{\alpha\beta} = \tilde{S}_{00}^{\alpha\beta} - \sum_{\mu\nu} \tilde{S}_{0\mu}^{\alpha\beta} \left(S^{-1} \right)_{\mu\nu}^{\alpha\beta} \tilde{S}_{0\nu}^{\alpha\beta} \qquad m^{\alpha} = \sum_{\alpha'\mu\nu} y^{\mu} \tilde{S}_{0\nu}^{\alpha'\alpha} \left(S^{-1} \right)_{\mu\nu}^{\alpha'\alpha} \tag{D.35}$$

D.2.1 RS interaction matrix

If we choose the matrix M_1 to be RS we have further simplifications:

$$\bar{Q} = \bar{q}^1 \mathbb{1}_Y + \bar{q}^0 \mathbb{O}_Y \tag{D.36}$$

$$S_{\mu\nu}^{\alpha\beta}(Q) = \frac{1}{\lambda_1} \left(\delta_{\alpha\beta} \bar{q}^1 K_{\mu\nu}^1 + (1 - \delta_{\alpha\beta}) \bar{q}^0 K_{\mu\nu}^0 \right) \tag{D.37}$$

And have to be evaluated at the saddle point:

$$\bar{q}_1^* = \frac{\gamma + \lambda_1}{\lambda_1 + Y\gamma} \qquad \bar{q}_0^* = \frac{\gamma}{\lambda_1 + Y\gamma} \tag{D.38}$$

$$
\begin{aligned}
S|_{\bar{Q}^*} &= \left(\mathbb{1}_P \otimes \mathbb{1}_Y + \frac{\bar{q}_0^*}{\bar{q}_1^*} K_0 (K_1)^{-1} \otimes \mathbb{0}_Y\right) \left(\frac{\bar{q}_1^*}{\lambda_1} K_1 \otimes \mathbb{1}_Y\right) \\
S^{-1} &= \left(\frac{\lambda_1}{\bar{q}_1^*} (K_1)^{-1} \otimes \mathbb{1}_Y\right) (C \otimes \mathbb{1}_Y + D \otimes \mathbb{0}_Y) \\
D &= -\left(\frac{q_1}{q_0} K_1 K_0^{-1} - (Y-1)\frac{q_0}{q_1} K_0 K_1^{-1} + (Y-2)\mathbb{1}_P\right)^{-1} \\
C &= \mathbb{1}_P + (Y-1)\frac{q_0}{q_1} K_0 K_1^{-1} D
\end{aligned}
$$

Here Σ and m inherit the RS structure from S:

$$
\begin{aligned}
\Sigma &= \Sigma^1 \mathbb{1}_Y + \Sigma^0 \mathbb{0}_Y \\
\Sigma_1|_{\bar{q}_1^*} &= \frac{1}{\lambda_1}\frac{\gamma+\lambda_1}{\lambda_1 + Y\gamma}\left(K^1_{00} - \frac{1}{\lambda_1}\frac{\gamma+\lambda_1}{\lambda_1+Y\gamma}\sum_{\mu\nu} K^1_{0\mu}\left(S^{-1}\right)^{\alpha\alpha}_{\mu\nu} K^1_{0\nu}\right) \\
\Sigma_0|_{\bar{q}_0^*} &= \frac{1}{\lambda_1}\frac{\gamma}{\lambda_1 + Y\gamma}\left(K^0_{00} - \frac{1}{\lambda_1}\frac{\gamma}{\lambda_1+Y\gamma}\sum_{\mu\nu} K^0_{0\mu}\left(S^{-1}\right)^{\alpha\beta}_{\mu\nu} K^0_{0\nu}\right) \\
m &= m_1 + (Y-1)m_0 \\
m_1 &= \frac{1}{\lambda_1}\frac{\gamma+\lambda_1}{\lambda_1+Y\gamma}\sum_{\mu\nu} y^\mu K^1_{0\nu}(S^{-1})^{\alpha\beta}_{\mu\nu} \\
m_0 &= \frac{1}{\lambda_1}\frac{\gamma}{\lambda_1+Y\gamma}\sum_{\mu\nu} y^\mu K^0_{0\nu}(S^{-1})^{\alpha\alpha}_{\mu\nu}
\end{aligned}
$$

The generalisation accuracy (at the saddle point):

$$
\epsilon_{acc} = H\left(\frac{-\text{sign}(y^0)\sqrt{Y}(m_1 + (Y-1)m_0)}{\sqrt{\Sigma_1 + (Y-1)\Sigma_0}}\right) \tag{D.39}
$$

APPENDIX E

Details of computations for locally connected networks with and without weight sharing

E.1 Dtailed derivation of the effective action for a shallow Locally Connected Network

Here I report the explicit computation of the effective action for one hidden layer LCNs [29]. For simplicity, we consider one-dimensional local interaction, but the calculation can be generalized to a d-dimensional setting. For LCNs, the output function reads:

$$f_{\text{LCN}}(x) = \frac{1}{\sqrt{N_c \lfloor N_0/S \rfloor}} \sum_{i=1}^{\lfloor N_0/S \rfloor} \sum_{a=1}^{N_c} v_i^a \sigma\left(h_i^a\right) \tag{E.1}$$

$$h_i^a(x) = \sum_{m=-\lfloor M/2 \rfloor}^{\lfloor M/2 \rfloor} \frac{W_{im}^a x_{Si+m}}{\sqrt{M}} \tag{E.2}$$

We recall the notation for the total number of weights in the last layer $N_1 = N_c\lfloor N_0/S \rfloor$. Note that in the special case $S = N_0$ we recover the FC architecture. Our objective is to construct the data-dependent partition function for this learning problem in the proportional limit:

$$Z = \int \prod_{i,a} dv_i^a \prod_{i,m,a} dW_{im}^a \, \exp\Big\{ -\frac{\lambda_1}{2}\|v\|^2 - \frac{\lambda_0}{2}\|W\|^2 - \frac{\beta}{2}\sum_{\mu=1}^{P}\left[y^\mu - f_{\text{LCN}}(x^\mu)\right]^2 \Big\}, \quad \text{(E.3)}$$

with f_{LCN} given in Eq. (E.2), and λ_0 and λ_1 respectively the Gaussian priors of the hidden and last layer. We introduce two sets of Dirac deltas, corresponding to the pre-activations of the hidden layer and the output of the network:

$$\prod_{i,a,\mu} \delta\Big(h_{ia}^\mu - \frac{1}{\sqrt{M}} \sum_{m=-\lfloor \frac{M}{2} \rfloor}^{\lfloor \frac{M}{2} \rfloor} W_{im}^a x_{Si+m}^\mu\Big), \qquad \prod_{\mu} \delta\Big(s^\mu - \frac{1}{\sqrt{N_1}} \sum_{i,a=1}^{\lfloor \frac{N_0}{S} \rfloor, N_c} v_i^a \sigma(h_{ia}^\mu)\Big). \quad \text{(E.4)}$$

Expressing the deltas directly in their standard Fourier representation, and averaging on the Gaussian weights we have:

$$\begin{aligned} Z = \int \prod_{i,a,\mu} dh_{ia}^\mu d\bar{h}_{ia}^\mu \prod_\mu ds^\mu d\bar{s}^\mu e^{-\frac{\beta}{2}\sum_\mu (y^\mu - s^\mu)^2 + i\sum_\mu s^\mu \bar{s}^\mu + i\sum_{\mu,i} h_{ia}^\mu \bar{h}_{ia}^\mu} \\ \times \prod_{i,a} \exp\left\{ -\frac{\left[\sum_{\mu=1}^P \bar{s}^\mu \sigma(h_{ia}^\mu)\right]^2}{2\lambda_1 N_1} \right\} \prod_{i,a} \exp\left\{ -\frac{\sum_{\mu,\nu=1}^P \bar{h}_{ia}^\mu C_{\mu\nu}^{ii} \bar{h}_{ia}^\nu}{2} \right\}. \end{aligned} \quad \text{(E.5)}$$

here the matrix $C_{\mu\nu}^{ii}$ is the diagonal part of the local covariance matrix defined in Eq. (2.30), i.e.

$$C_{\mu\nu}^{ii} = \frac{1}{\lambda_0 M} \sum_{m=-\lfloor M/2 \rfloor}^{\lfloor M/2 \rfloor} x_{Si+m}^\mu x_{Si+m}^\nu. \quad \text{(E.6)}$$

The integrals on the h-$\bar{h}$ variables can be factorised over the patch indices i and channel one a. Performing the (Gaussian) integrations over the $\bar{h}$ variable leads to the following equation

for the partition function

$$Z = \int \prod_{\mu} ds^{\mu} d\bar{s}^{\mu} e^{-\frac{\beta}{2}\sum_{\mu=1}^{P}(y^{\mu}-s^{\mu})^2 + i\sum_{\mu=1}^{P} s^{\mu}\bar{s}^{\mu}} \times \left\{ \prod_{i} \int d^P h_i \, P_1^i(\{h^{\mu}\}) e^{-\frac{1}{2\lambda_1 N_1}[\sum_{\mu=1}^{P} \bar{s}^{\mu}\sigma(h_i^{\mu})]^2} \right\}^{N_c}, \tag{E.7}$$

$$P_1^{(i)}(\{h^{\mu}\}) \equiv \frac{e^{-\frac{1}{2}h_i^{\top}(C^{ii})^{-1}h_i}}{\sqrt{(2\pi)^P \det C^{ii}}}. \tag{E.8}$$

To proceed in the calculation, we introduce a new set of variables q_i through Dirac's δ identities. In this way the quantity in curly brakets in Eq. (E.7) becomes:

$$\prod_{i} \int dq_i e^{-\frac{1}{2}q_i^2} P(q_i)\,, \quad P(q_i) = \int d^P h_i P_1^i(\{h^{\mu}\}) \delta\Big(q_i - \frac{1}{\sqrt{\lambda_1 N_1}} \sum_{\mu=1}^{P} \bar{s}^{\mu}\sigma(h_i^{\mu})\Big) \tag{E.9}$$

Similarly to [95], we perform a Gaussian approximation on the limiting distribution of $P(q_i)$, heuristically justified by the Breuer-Major theorem [99]:

$$P(q_i) \to \mathcal{N}_{q_i}(0, Q_i), \tag{E.10}$$

where the variances are:

$$Q_i(\bar{s}, C^i) = \frac{1}{\lambda_1 N_1} \sum_{\mu\nu=1}^{P} \bar{s}^{\mu} \Big[\int d^P h P_1^i(\{h^{\mu}\}) \sigma(h^{\mu})\sigma(h^{\nu}) \Big] \bar{s}^{\nu} \equiv \frac{1}{\lambda_1 N_1} \sum_{\mu\nu=1}^{P} \bar{s}^{\mu} K^i_{\mu\nu} \bar{s}^{\nu}\,, \tag{E.11}$$

here $K^i_{\mu\nu} \equiv K^{ii}_{\mu\nu}$ is a compact notation for the diagonal part (in the spatial indices) of the kernel matrix defined in (2.29). After performing the (Gaussian) integrals in the q_i variables, we have:

$$\Big\{ \prod_{i} (1 + Q_i(\bar{s}, C^i))^{-\frac{1}{2}} \Big\}^{N_c} = e^{-\frac{N_c}{2}\sum_i \log(1+Q_i(\bar{s}, C^i))}. \tag{E.12}$$

We need one last delta to handle the $\bar{s}$ dependency in $Q_i(\bar{s}, C^i)$):

$$1 = \int dQ_i \, \delta\Big(Q_i - \frac{1}{\lambda_1 N_1} \sum_{\mu\nu=1}^{P} \bar{s}^\mu K^i_{\mu\nu} \bar{s}^\nu\Big). \tag{E.13}$$

In this way we are able to perform the Gaussian integration on the $\bar{s}$ variable, after a rescaling $\bar{Q} \to -i\frac{N_c}{2}\bar{Q}$, obtaining:

$$\begin{aligned} Z = \int \prod_i dQ_i d\bar{Q}_i \prod_\mu ds^\mu \, \exp(-\frac{\beta}{2}\sum_{\mu=1}^{P}(y^\mu - s^\mu)^2) \\ \times \exp\left(-\frac{1}{2} s^\top \left[K^{(\mathrm{R})}_{\mathrm{LCN}}\right]^{-1} s - \frac{1}{2}\log\det\left[K^{(\mathrm{R})}_{\mathrm{LCN}}\right] - \frac{N_c}{2}\sum_i \left(Q_i\bar{Q}_i - \log(1+Q_i)\right)\right), \end{aligned} \tag{E.14}$$

where we have identified the renormalized kernel for the LCN:

$$\left[K^{(\mathrm{R})}_{\mathrm{LCN}}(\bar{Q})\right]_{\mu\nu} \equiv \frac{1}{\lfloor N_0/S \rfloor} \sum_{i=1}^{\lfloor N_0/S \rfloor} \bar{Q}_i K^i_{\mu\nu} \, . \tag{E.15}$$

which contains only the diagonal elements of the local kernel matrix defined in Eq. (4.54). Finally, we perform this last Gaussian integral and we are left with a partition function which depends only on Q and $\bar{Q}$:

$$Z = \int \prod_i dQ_i d\bar{Q}_i e^{-\frac{N_c}{2} S_{\mathrm{LCN}}(Q,\bar{(Q)}}, \tag{E.16}$$

where the LCN effective action S_{LCN} is given by:

$$\begin{aligned} S_{\mathrm{LCN}} \equiv -\sum_i (Q_i\bar{Q}_i - \log(1+Q_i)) + \\ + \frac{\alpha_c}{P}\mathrm{Tr}\log\beta\Big(\frac{\mathbb{1}_P}{\beta} + K^{(\mathrm{R})}_{\mathrm{LCN}}\Big) + \frac{\alpha_c}{P} y^\top \Big(\frac{\mathbb{1}_P}{\beta} + K^{(\mathrm{R})}_{\mathrm{LCN}}\Big)^{-1} y. \end{aligned} \tag{E.17}$$

where $\mathbb{1}_P$ denotes the $P \times P$ identity matrix. Note that this action shares the same functional form as the one for FCN found in [95].

E.2 Detailed derivation of the effective action for a shallow CNN

In this section we address the case of shallow CNN architectures, where there is combination of locality and weight sharing. The output of a CNN reads:

$$f_{\text{CNN}}(x^\mu) = \frac{1}{\sqrt{N_1}} \sum_{i=1}^{\lfloor N_0/S \rfloor} \sum_{a=1}^{N_c} v_i^a \sigma \left(\sum_{m=-\lfloor M/2 \rfloor}^{\lfloor M/2 \rfloor} \frac{W_m^a x_{Si+m}^\mu}{\sqrt{M}} \right), \tag{E.18}$$

Similarly to the LCN case, we need two sets of deltas, one for the preactivations and one for the outputs, that we directly insert through their standard Fourier representation:

$$\begin{aligned} Z = \int \prod_{i,a,\mu} dh_{ia}^\mu d\bar{h}_{ia}^\mu \prod_\mu ds^\mu d\bar{s}^\mu e^{-\frac{\beta}{2}\sum_{\mu=1}^P (y^\mu - s^\mu)^2 + i\sum_{\mu=1}^P s^\mu \bar{s}^\mu + i\sum_{\mu ia} h_{ia}^\mu \bar{h}_{ia}^\mu} \\ \prod_a \exp\left\{ -\frac{1}{2\lambda_1 N_1} \sum_i \left[\sum_{\mu=1}^P \bar{s}^\mu \sigma(h_{ia}^\mu) \right]^2 \right\} \prod_a \exp\left\{ -\frac{1}{2} \sum_{i,j} \sum_{\mu,\nu} \bar{h}_{ia}^\mu C_{\mu\nu}^{ij} \bar{h}_{ja}^\nu \right\}. \end{aligned} \tag{E.19}$$

where we have used the definition (E.6) of the local covariance matrix. Factorizing over the channel index, and integrating the Gaussian $\bar{h}_i$ the partition function reads:

$$\begin{aligned} Z = \int \prod_\mu ds^\mu d\bar{s}^\mu e^{-\frac{\beta}{2}\sum_{\mu=1}^P (y^\mu - s^\mu)^2 + i\sum_{\mu=1}^P s^\mu \bar{s}^\mu} \\ \times \left\{ \int \prod_{\mu i} dh_i^\mu \, \mathcal{N}_h(0, C) e^{-\frac{1}{2\lambda_1 N_1}\sum_i [\sum_{\mu=1}^P \bar{s}^\mu \sigma(h_i^\mu)]^2} \right\}^{N_c}. \end{aligned} \tag{E.20}$$

where the notation $\mathcal{N}_h(0, C)$ indicates a multivariate normalized Gaussian on the h variables, with zero mean and covariance matrix $C_{\mu\nu}^{ij}$. In order to deal with the non linearity term, we define a collection of $N_1 = \lfloor N_0/S \rfloor$ deltas:

$$1 = \int \prod_i dq_i \, \delta\Big(q_i - \frac{1}{\sqrt{\lambda_1 N_1}} \sum_{\mu=1}^P \bar{s}^\mu \sigma(h_i^\mu) \Big). \tag{E.21}$$

Here again, we assume that the joint distribution of the variables q_i can be approximated by a multivariate Normal distribution with 0 mean and covariance matrix Q with elements:

$$Q_{ij}(\bar{s}, C) \equiv \langle q_i q_j \rangle_h = \frac{1}{\lambda_1 N_1} \sum_{\mu,\nu} \bar{s}^\mu \langle \sigma(h_i^\mu)\sigma(h_j^\nu)\rangle_h \bar{s}^\nu \equiv \frac{1}{\lambda_1 N_1} \sum_{\mu,\nu} \bar{s}^\mu K^{ij}_{\mu\nu} \bar{s}^\nu, \tag{E.22}$$

Note that this would be heuristically justified by a multivariate version of the Breuer-Major theorem. The quantity in curly brackets is again a multidimensional Gaussian integral on the q_i variables:

$$\begin{aligned}\Big\{ \dots \Big\}^{N_c} =& \left\{ \int \prod_i dq_j \, \frac{e^{-\frac{1}{2}\sum_{ij} q_i(\delta_{ij}+Q^{-1}_{ij})q_j}}{\sqrt{\det 2\pi Q}} \right\}^{N_c} = (\det(\mathbb{1}+Q))^{-\frac{N_c}{2}} \\ &= e^{-\frac{N_c}{2}\operatorname{Tr}\log(\mathbb{1}+Q)}\end{aligned} \tag{E.23}$$

To deal with the implicit dependence of Q on $\bar{s}$, we introduce one last family of deltas for the local variables Q_{ij}:

$$1 = \int dQ_{ij}\, \delta\Big(Q_{ij} - \frac{1}{\lambda_1 N_1} \sum_{\mu\nu=1}^{P} \bar{s}^\mu K^{ij}_{\mu\nu} \bar{s}^\nu\Big), \tag{E.24}$$

that allow to perform the integration over $\bar{s}^\mu$ after the transformation $\bar{Q}_{ij} \to -\frac{iN_c}{2}\bar{Q}_{ij}$:

$$\begin{aligned} Z = \int \prod_{ij} dQ_{ij} d\bar{Q}_{ij} \prod_\mu ds^\mu \, \exp\left(\frac{N_c}{2} \sum_{ij} Q_{ij}\bar{Q}_{ij} + \beta \sum_{\mu=1}^{P} s^\mu y^\mu - \frac{\beta}{2} \sum_{\mu=1}^{P} y^\mu y^\mu \right) \\ \times \exp\left(-\frac{N_c}{2} \operatorname{Tr}\log(\mathbb{1}+Q) - \frac{1}{2}\log\det\Big[K^{(\mathrm{R})}_{\mathrm{CNN}}\Big] + \right. \qquad \text{(E.25)} \\ \left. -\frac{1}{2}\sum_{\mu\nu=1}^{P} s^\mu \Big(\beta\delta_{\mu\nu} + \Big[K^{(\mathrm{R})}_{\mathrm{CNN}}\Big]^{-1}_{\mu\nu}\Big) s^\nu \right), \qquad \text{(E.26)} \end{aligned}$$

where $\Big[K^{(\mathrm{R})}_{\mathrm{CNN}}\Big]_{\mu\nu}$ is the renormalized CNN kernel defined in (4.54). After performing the last Gaussian integral on s^μ, the partition function is expressed through an effective action

$S_{\text{CNN}}(Q,\bar{Q})$ that only depends on the order parameters Q_{ij}:

$$Z = \int \prod_{ij} dQ_{ij} d\bar{Q}_{ij} e^{-\frac{N_c}{2} S_{\text{CNN}}(Q,\bar{Q})}, \tag{E.27}$$

$$S_{\text{CNN}}(Q,\bar{Q}) \equiv -\sum_{ij} Q_{ij}\bar{Q}_{ij} + \text{Tr}\log(\mathbb{1}_{\lfloor \frac{N_0}{S} \rfloor} + Q) + \tag{E.28}$$

$$+ \frac{\alpha_c}{P}\text{Tr}\log\beta\Big(\frac{\mathbb{1}_P}{\beta} + K^{(\text{R})}_{\text{CNN}}\Big) + \frac{\alpha_c}{P} y^\top \Big(\frac{\mathbb{1}_P}{\beta} + K^{(\text{R})}_{\text{CNN}}\Big)^{-1} y, \tag{E.29}$$

Note that, differently from the previous section, all the information of the matrix Q is present at the level of the effective action.

E.2.1 Saddle point equations from the CNN effective action

In this section we explicitly derive the saddle point equations of the effective CNN action given in Eq. (E.29), that is:

$$\frac{\partial}{\partial Q_{ij}} S_{\text{CNN}}(Q,\bar{Q}) = 0 \tag{E.30}$$

$$\frac{\partial}{\partial \bar{Q}_{ij}} S_{\text{CNN}}(Q,\bar{Q}) = 0 \tag{E.31}$$

Recalling that $\text{Tr}\log(\mathbb{1}+Q) \equiv \log\det(\mathbb{1}+Q)$ and $\frac{\partial}{\partial A_{ij}}\det A = \text{adj}(A)_{ij}$, Eq. (E.30) reads:

$$0 = -\bar{Q}_{ij} + (\text{adj}(\mathbb{1}+Q))_{ij}(\det(\mathbb{1}+Q))^{-1}. \tag{E.32}$$

Using the identity $A^{-1} = \text{adj}(A)(\det A)^{-1}$ we find

$$\bar{Q}_{ij} = (\mathbb{1}+Q)^{-1}_{ij}, \tag{E.33}$$

which holds for each element of $\bar{Q}$ and so $\bar{Q} = (\mathbb{1}+Q)^{-1}$.

To compute the derivative in (E.30), one should observe that $\Big(\frac{\mathbb{1}}{\beta} + K^{(\text{R})}_{\text{CNN}}\Big)$ can be thought as a $P \times P$ matrix field of the matrix variable $\bar{Q}$. We use of the following identity for a generic

matrix $A = A(x)$:

$$\partial_{x_i} \det A = (\det A) \cdot \mathrm{Tr}(A^{-1} \partial_{x_i} A)\,, \tag{E.34}$$

and obtain:

$$0 = -Q_{ij} + \frac{\alpha_c}{P} \mathrm{Tr}\left[\left(\frac{\mathbb{1}}{\beta} + K^{(\mathrm{R})} \right)^{-1} K^{ij} \right] \tag{E.35}$$

$$- \frac{\alpha_c}{P} \sum_{\mu\rho\lambda\nu} y_\mu \left(\frac{\mathbb{1}}{\beta} + K^{(\mathrm{R})} \right)^{-1}_{\mu\rho} K^{ij}_{\rho\lambda} \left(\frac{\mathbb{1}}{\beta} + K^{(\mathrm{R})} \right)^{-1}_{\lambda\nu} y_\nu, \tag{E.36}$$

which reduces to Eq. (5.5) in the zero temperature limit.

APPENDIX F

Numerical experiments

F.1 Details of numerical simulations

F.1.1 Sampling from the Bayesian posterior

To ensure convergence of the posterior weights distribution to the Gibbs ensemble, the networks are trained using a discretised Langevin dynamics, similarly to what is done in [9, 11]. At each training step t the parameters $\theta = \{W^\ell, v\}$ are updated according to:

$$\theta(t+1) = \theta(t) - \eta \nabla_\theta \mathcal{L}(\theta(t)) + \sqrt{2T\eta}\epsilon(t) \tag{F.1}$$

where $T = 1/\beta$ is the temperature, η is the learning rate, $\epsilon(t)$ is a white Gaussian noise vector with entries drawn from a standard normal distribution, and the loss is the one defined in equation (4.2).

F.1.2 Experiments with FC networks

Numerical experiments are performed with deep fully-connected architectures trained on two regression tasks in computer vision. In particular we use the 0 and 1 classes of the MNIST and CIFAR10 datasets, which for the latter correspond to the labels "cars" and "planes". Examples from CIFAR10 are coarse grained to $N_0 = 28 \times 28$ pixels and converted to grayscale.

To test our theory in the zero-mean activation function case, we used the Erf function, for which the NNGP kernel can be computed analytically [117]:

$$K^{\text{Erf}}_{\mu\nu}(C) = \frac{2}{\pi} \arcsin\left(\frac{2C_{\mu\nu}}{\sqrt{(1+2C_{\mu\mu})\,(1+2C_{\nu\nu})}} \right) . \tag{F.2}$$

In Fig. 5.3 we train networks with $\sigma = \text{ReLU}$. The kernel can be computed analytically also in this case [109] and reads:

$$K^{\text{ReLU}}_{\mu\nu}(C) = \sqrt{C_{\mu\mu}C_{\nu\nu}}\,\kappa\left(\frac{C_{\mu\nu}}{\sqrt{C_{\mu\mu}C_{\nu\nu}}} \right) , \tag{F.3}$$
$$\kappa(x) = \frac{1}{2\pi}\left[x(\pi - \arccos(x)) + \sqrt{1-x^2} \right] .$$

We employ $T = \eta = 10^{-3}$ throughout all the experiments. This is sufficient to approximate the $T = 0$ dynamics in the regime we are considering. This dynamics requires $10^5/10^6$ steps to reach thermalisation, depending on the sizes of the dataset and network. We extract the generalisation loss within a single run: after the train error has reached its minimum and the test loss is thermalised, we average test loss values every $10^3/10^4$ epochs (depending again on the magnitude of P, N_ℓ). For the sake of completeness, we report the best test accuracy achieved on both datasets by 1HL architectures: 0.86 on CIFAR10 with $P = 3000$ and $\lambda_1 = 1000$, 0.999 on MNIST with the same Gaussian prior and $P = 1000$. The train accuracy is always 1. The code used to perform experiments with FC networks, compute theory predictions and analyze data is available at: `https://github.com/rpacelli/FC_deep_bayesian_networks` [118].

F.1.3 Experiments with 1d convolutions

The finite-width analysis presented in Fig. 5.5 was carried out training a one hidden layer network with 1d convolutional filters on a synthetic random dataset, whose elements have entries sampled from a Gaussian distribution $\mathcal{N}(0,1)$, with labels given by a linear teacher function with unitary weights $t = \{1 \dots 1\}$:

$$y^{\mu} = \frac{1}{2}\left(1 + \text{sign}\left(t \cdot x^{\mu}\right)\right). \tag{F.4}$$

The network implements exactly the function reported in Eq. (4.48), with activation function $\sigma(x) = \tanh(x)$. The FC architecture is retrieved setting both the filter size M and the stride S equal to the input dimension $N_0 = M = S$. This correctly implements the function given in Eq. (4.5). Here the networks are trained using full-batch gradient descent, with ADAM optimizer, implemented with TensorFlow (TF). We train using a large value of the last layer Gaussian prior λ_1, until the loss reaches a value of $O(10^{-10})$. We employed a scheduler for the learning rate, reducing its value from 10^{-3} to 10^{-7} with the so-called ReduceLROnPlateau scheduler of TF. The experiments shown in Fig. (5.5) were performed fixing the value of $\alpha_1 = P/N_1 = 1$ with increasing values of N_1. The input size is set to $N_0 = 6400$ and we choose non-overlapping convolutional filters, taking the size of the mask and the stride equal to $M = S = 400$. We built a statistical sample of $O(10^2)$ networks, trained independently, over which we average each result.

F.1.4 Experiments with 2d convolutions

The results shown in Fig. 5.6 are obtained training a one hidden layer architecture with Erf activation and $2d$ non-overlapping convolutional filters on the CIFAR10 binary task discussed above. This is achieved setting the stride S to be equal to the filter mask size M. To avoid information loss, we choose M to be an integer divisor of the linear input size $d = 28$. To ensure convergence of the posterior weights distribution to the Gibbs ensemble, we train our networks using a discretized Langevin dynamics, similarly to what is done in [9, 11, 95]. At

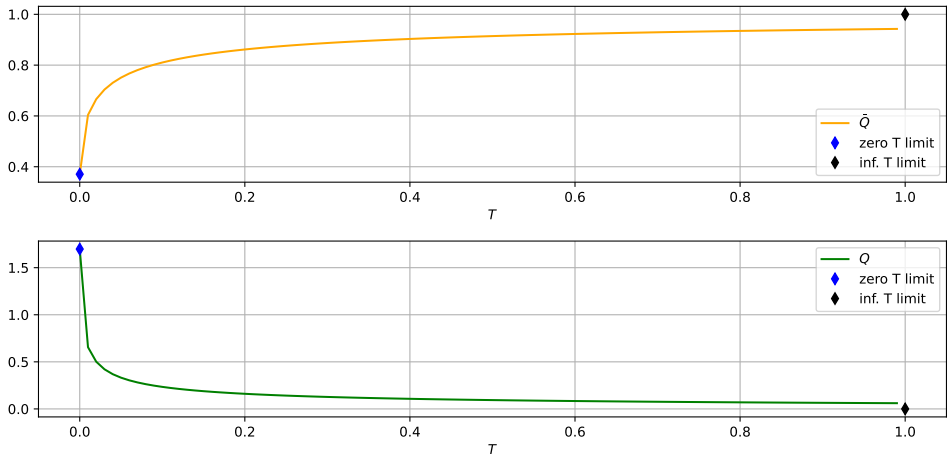


Figure F.1 –Numerical solution for the order parameters Q, $\bar{Q}$ as a function of the temperature T.

each training step t the parameters $\theta = \{W, v\}$ are updated according to:

$$\theta(t+1) = \theta(t) - \eta \nabla_\theta \mathcal{L}(\theta(t)) + \sqrt{2T\eta}\epsilon(t) \tag{F.5}$$

where $T = 1/\beta$ is the temperature, η is the learning rate, $\epsilon(t)$ is a white Gaussian noise vector with entries drawn from a standard normal distribution, and the loss is the one defined in equation (4.2). We employ $T = \eta = 2 \cdot 10^{-3}$ throughout all these experiments. This is sufficient to approximate the $T = 0$ dynamics in the regime we are considering. This dynamics requires $\sim 10^6$ steps to reach thermalization, in particular we run the experiment for $5 \cdot 10^6$ epochs. When possible, we extract the generalization loss within a single run: after the train error has reached its minimum and the test loss is thermalized, we average test loss values every 10^3 epochs. In the case of FC architecture in Fig. 5.6, we averaged over $n = 3$ samples to reduce the error.

F.2 Numerical issues in sampling from the Bayesian posterior

Obtaining a perfect agreement between theory and simulations when sampling from a Bayesian posterior (especially in the zero temperature limit) is prevented by a number of technical numerical issues presented in the following.

1. Finite-size effects certainly play a role in explaining the small mismatch between theory and experiment. To address this point, we are currently performing high-precision numerical simulations with fixed $\alpha = P/N_1$ and increasing values of N_1 and P.

2. The $T \to 0$ limit, which corresponds to perfect interpolation of the dataset and is the only case in which the saddle point equations can be solved analytically, was the most logic to address for starting, but turns out to be very hard to simulate. This is clear from some preliminary work we are doing, where we numerically solve the saddle point equations at generic T for the saddle point variables Q, $\bar{Q} = f(Q)$. We find that the function $Q(T)$ changes rapidly for small temperatures, as is shown in Figure F.1.

3. At $T = 0.001$, the autocorrelation time of the simulation is already very large, taking as little as $5 \cdot 10^6$ epochs to thermalize. As the temperature is decreased, the autocorrelation time increases, and we need hundreds of thousands of epochs to gain satisfactory statistics.

4. The effect of a finite learning rate η has to be taken into account as well. From our preliminary results, we empirically observe that finite-η effects are larger at higher temperature. The standard way to take into account finite-η effects is to perform the extrapolation to $\eta \to 0$ simulating different learning rates.

5. Computing the theory in the case of $L > 1$ networks requires to numerically minimize a complex nested saddle-point functional of the variables $\bar{Q}_\ell$. We are currently working on a numerical routine to efficiently perform this task.

•