Lectures on Deep Learning Theory

Boris Hanin
Princeton ORFE

May 20, 2024

Contents

1 Neural Networks: An Overview

1.1 Introduction

At a high level, a neural network is a parameterized family of functions. In these notes we will deal primarily with the simplest so-called fully connected neural networks (see §1.1.1 below). In practice, neural networks are most commonly used for supervised learning, i.e. finding a good approximation to an unknown function f from a dataset consisting of its (possibly corrupted) values at either deterministic or randomly sampled points. We will review how this works in §1.2. We then formulate and discuss several overarching questions that have motivated much of the recent theoretical analysis of neural networks in §1.3.

1.1.1 Definition

Definition 1.1.1 (Fully Connected Network (e.g. [Han23]))**.** *Fix a positive integer L as well as $L+2$ positive integers $N_0, \dots, N_{L+1}$ and a function $\sigma : \mathbb{R} \to \mathbb{R}$. A fully connected depth L neural network with input dimension N_0, output dimension N_{L+1}, hidden layer widths $N_1, \dots, N_L$, and non-linearity σ is any function that maps inputs $x_\alpha \in \mathbb{R}^{N_0}$ to outputs $z_\alpha^{(L+1)} \in \mathbb{R}^{N_{L+1}}$ as follows:*

$$z_\alpha^{(\ell)} = \begin{cases} W^{(1)} x_\alpha + b^{(1)}, & \ell = 1 \\ W^{(\ell)} \sigma(z_\alpha^{(\ell-1)}) + b^{(\ell)}, & \ell = 2, \dots, L+1 \end{cases}, \tag{1.1.1}$$

where $W^{(\ell)} \in \mathbb{R}^{N_\ell \times N_{\ell-1}}$ are matrices, $b^{(\ell)} \in \mathbb{R}^{N_\ell}$ are vectors, and σ applied to a vector is shorthand for σ applied to each component.

The parameters $L, N_0, \dots, N_{L+1}$ are called the *network architecture*, and $z_\alpha^{(\ell)} \in \mathbb{R}^{N_\ell}$ is called the *vector of pre-activations at layer ℓ* corresponding to input x_α. We will sometimes write

$$z_\alpha^{(\ell)} = z_\alpha^{(\ell)}(\theta)$$

when we wish to emphasize the dependence of the pre-activations (or network outputs if $\ell = L+1$) on the network parameters θ, which are simply a flattened list of the entries $W_{ij}^{(\ell)}$ of the weight matrices and components $b_i^{(\ell)}$ of the bias vectors. We will also sometimes write

$$z_\alpha^{(\ell)} = \left(z_{i;\alpha}^{(\ell)},\, i = 1, \dots, N_\ell \right)$$

where we wish to speak about the individual components of the pre-activations. In this notation, the recursion (1.1.1) reads

$$z_{i;\alpha}^{(\ell)} = \begin{cases} \sum_{j=1}^{N_0} W_{ij}^{(1)} x_{i;\alpha} + b_i^{(1)}, & \ell = 1 \\ \sum_{j=1}^{N_\ell} W_{ij}^{(\ell)} \sigma(z_{j;\alpha}^{(\ell-1)}) + b_i^{(\ell)}, & \ell = 2, \dots, L+1 \end{cases}. \tag{1.1.2}$$

1.1.2 Two Examples of Functions Computed by a Neural Network

In this section, we give two brief examples of functions computed by ReLU networks i.e. fully connected networks with the most popular non-linearity used in practice:

$$\sigma(t) = \mathrm{ReLU}(t) := \max\{0, t\}.$$

The first example is the following:

$$x \quad \mapsto \quad \sigma\left(\begin{pmatrix} 2 \\ -4 \end{pmatrix}^T \sigma\left(\begin{pmatrix} 1 \\ 1 \end{pmatrix} x + \begin{pmatrix} 0 \\ -1/2 \end{pmatrix}\right)\right) =: f(x).$$

A quick computation shows that this is the triangle or hat function:

$$f(x) = \begin{cases} 2x, & x \in [0, 1/2] \\ 1 - 2x, & x \in [1/2, 1] \\ 0, & x \in (-\infty, 0) \cup (1, \infty) \end{cases},$$

which plays a fundamental role in signal processing. The second example is more broad. Consider all one hidden layer ReLU networks with input and output dimensions equal to 1 (i.e. $L = 1, N_0 = N_2 = 1$):

$$z(x) = b^{(2)} + \sum_{i=1}^{N_1} W_i^{(2)} \sigma\left(W_i^{(1)} x + b_i^{(1)}\right), \qquad b_i^{(1)}, W_i^{(1)}, W_i^{(2)}, b^{(2)} \in \mathbb{R}. \tag{1.1.3}$$

The key point is the following

Lemma 1.1.2. *The space of functions obtained by varying $W_i^{(1)}, b_i^{(1)}, W_i^{(2)}, b^{(2)}$ in (1.1.3) contains all continuous piecewise linear functions with at most $N_1 - 1$ breakpoints (i.e. points of discontinuity for the derivative) and is contained in the space of all continuous piecewise linear functions with N_1 breakpoints.*

Proof Sketch. The idea is that the the i-th neuron $x \mapsto W_i^{(2)} \sigma\left(W_i^{(1)} x + b_i^{(1)}\right)$ is continuous and piecewise linear with a single breakpoint at

$$\xi_i := -b_i^{(1)} / W_i^{(1)}.$$

The remainder of the argument is left as an exercise. □

1.2 Typical Use

The usual process by which one uses a neural network to "learn" from data can be roughly divided into four steps:

1. **Data Acquisition.** Collect a dataset $\mathcal{D} = \{(x_\alpha, y_\alpha))\}_{\alpha=1}^{|\mathcal{D}|}$ consisting of potentially noisy observations of an unknown function:

$$y_\alpha = f(x_\alpha) + \epsilon(x_\alpha), \qquad \epsilon(x_\alpha) \sim \mathcal{N}(0, \sigma_\epsilon^2). \tag{1.2.1}$$

2. **Architecture Selection.** Choose an architecture, i.e. $L, N_1, \ldots, N_L, \sigma$, which specifies the form of the computation $x_\alpha \mapsto z_\alpha^{(L+1)}(\theta)$ but does not determine the trainable parameters $\theta = \left\{W^{(\ell)}, b^{(\ell)},\, 1 \leq \ell \leq L+1\right\}$.

3. **Initialization.** Randomly select the weights and biases to produce a random θ_0 and hence a random function $z_\alpha^{(L+1)}(\theta_0)$.

4. **Optimization.** Obtain a final setting of parameters θ_* by repeatedly updating θ using a first order optimization method such as gradient descent

$$\theta_{t+1} = \theta_t - \nabla_\theta \mathcal{L}(\theta)\Big|_{\theta=\theta_t}$$

on an appropriate empirical loss

$$\mathcal{L}_{\mathcal{D}}(\theta) = \frac{1}{|\mathcal{D}|}\sum_{\alpha=1}^{|\mathcal{D}|} \ell(y_\alpha, z_\alpha^{(L+1)}(\theta)) \tag{1.2.2}$$

where ℓ measure how close the current predictions are to their the target y (e.g. $\ell(y,z) = (y-z)^2$).

The goal of any supervised learning procedure, such as Steps 1-4 above, is not only to (approximately) fit the data by asking that

$$\mathcal{L}_{\mathcal{D}}(\theta_*) \text{ is small} \tag{1.2.3}$$

but also to generalize (i.e. extrapolate) by asking that

$$\ell(y, z^{(L+1)}(x;\theta_*)) \text{ is small on average or with high probability over the law } (x,y). \tag{1.2.4}$$

1.3 Big Questions and Small Intuitions

A number of specific tricks are employed in practice for steps $1-4$ to yield a good approximation in the sense of (1.2.3) and (1.2.4). However, that this process works well at all is somewhat surprising and motivates a number of important theoretical questions about neural networks. We discuss several such questions in the sections below.

1.3.1 Q1. Success of non-convex optimization

As explained in §1.2 training a neural network $z(x;\theta)$ in practice almost always consists of minimizing an empirical loss

$$\mathcal{L}_{\mathcal{D}}(\theta) = \frac{1}{|\mathcal{D}|}\sum_{(x,y)\in\mathcal{D}} \ell(y, z(x;\theta)) \tag{1.3.1}$$

using a first order method such a gradient descent

$$\theta_{t+1} = \theta_t - \eta \nabla_\theta \mathcal{L}(\theta)\big|_{\theta=\theta_t}. \tag{1.3.2}$$

One should expect such a greedy local search to converge to a stationary point of the loss (i.e. a point where $\nabla_\theta \mathcal{L}(\theta) = 0$). Thus, with high probability or perhaps with probability 1 in the presence of optimization noise, it therefore seems sensible to predict at large T that θ_T will be close to a *local minimum* of $\mathcal{L}$ [JGN$^+$17, **?**]. However, in practice, what we find is that θ_T is close to a *global minimum* of $\mathcal{L}$.

Explaining under what assumptions this statement can be made rigorous is an important open problem in modern machine learning. However, there is a simple intuition for at least one important mechanism driving the success of non-convex optimization. Namely, neural networks $z(x;\theta)$ in practice are almost always overparameterized in the sense that

$$\#\text{parameters} \quad \gg \quad \#\text{ training datapoints}^1. \tag{1.3.3}$$

The key observation is that while $\mathcal{L}(\theta)$ is a complex function of θ, it is a simple (e.g. convex) function in the variables

$$z(\mathcal{D};\theta) := (z(x;\theta), \quad (x,y) \in \mathcal{D})$$

that record the outputs on the training dataset. Moreover, precisely because of (1.3.3), a heuristic dimension-counting argument suggests

$$\text{the change of variables} \qquad \theta \quad \mapsto \quad z(\mathcal{D};\theta) \qquad \text{is a surjective submersion} \tag{1.3.4}$$

locally near almost every θ. By definition, we thus expect that the Jacobian $D_\theta z(\mathcal{D};\theta) \in \mathbb{R}^{\#\text{parameters}\times|\mathcal{D}|}$ of this map has full rank for most values of θ. Thus, since $\mathcal{L}(\theta)$ depends on θ only via $z(\mathcal{D};\theta)$, we obtain

$$\nabla_\theta \mathcal{L}(\theta) = 0 \qquad \Longleftrightarrow \qquad D_\theta z(\mathcal{D};\theta) \nabla_z \mathcal{L}(z)\big|_{z=z(\mathcal{D};\theta)} = 0 \qquad \Longleftrightarrow \qquad \nabla_z \mathcal{L}(z)\big|_{z=z(\mathcal{D};\theta)} = 0.$$

Since $\mathcal{L}$ is a simple (e.g. convex) function of $z(\mathcal{D};\theta)$, we conclude that a stationary point of $\mathcal{L}$ with respect to θ necessarily corresponds to a global minimum of $\mathcal{L}$. A cartoon of this situation is illustrated in Figure **??**.

Brief Bibliography on Success of Non-Convex Optimization

Many excellent articles have been written on the subject of optimization with neural networks. Below is a far from representative list of those pertaining to the success of non-convex optimization:

- **Linear Networks.** Neural networks in which the non-linearity $\sigma : \mathbb{R} \to \mathbb{R}$ is the identity are called linear networks. The articles [BH89, Kaw16, LK17] show that, under some technical assumptions, all local minima of the empirical ℓ_2-loss over a fixed training dataset are global minima.

- **NTK Regime.** As we discuss in §2.2, under certain parameterizations, neural networks with a fixed depth are close to linear models at large width. Hence, the empirical ℓ_2 loss over a training dataset is convex, which allows one to guarantee convergence to a global minimum. The first work of this kind appears to be [DLT$^+$18] in the special case

[1] Actually in the setting of modern LLMs one often has more training datapoints than parameters. However, these datapoints are far from iid, with many very similar examples. So in the heuristic outlined below one should imagine using an appropriate (and still not understood) notion of *effective* number of datapoints.

of one layer ReLU networks. This was then recognized as a general phenomenon for any non-linearity and any fixed depth in the seminal work [JGH18]. This perspective was then extended and refined in [ADH+19b] and [LZB22a]. Many other excellent articles along these lines have been written, e.g. [DZPS19, OS20, LZB22b].

1.3.2 Q2. Generalization with overparameterized interpolation

A cornerstone of traditional statistical learning theory is the bias-variance tradeoff. Roughly it says that although complex models (e.g. neural networks with many parameters) can interpolate data in the sense of (1.2.3), such models are so flexible that they will fit not only the signal, $f(x)$, but also the noise $\epsilon(x)$ inherent in the training data. The variance of predictors learned by such models will therefore be large, causing them to make unreliable predictions on new data. Overparameterized neural networks are indeed more than capable of perfectly fitting noisy data (and even random noise). But, when trained on real data, they often make good predictions on unseen data [ZBH+17]. Understanding why this is possible raises two important questions in deep learning theory:

- **Implicit Bias.** Under the overparameterization condition (1.3.3) there are infinitely many θ that minimize $\mathcal{L}$ (e.g. for which $z(x;\theta) = y$ for all $(x, y) \in \mathcal{D}$). Some of these θ correspond to functions $z(x;\theta)$ that make terrible predictions on new values of x. This is true even for noiseless data (i.e. with $\sigma_\epsilon^2 = 0$ in (1.2.1)). However, which θ is chosen depends very much on the learning algorithm and the learning algorithms used in practice seem to be able to select, among the sea of minimizers of $\mathcal{L}$, "good" values of θ. Understanding how the details of the learning algorithm (usually a variant of SGD) determine the properties of the learned minimizer of $\mathcal{L}$ is sometimes referred to as understanding the implicit bias or implicit regularization. Just below, we illustrate perhaps the simplest example of the implicit bias of gradient descent, in which appropriately initialized SGD returns the minimal ℓ_2-norm interpolant in ordinary least squares regression.

- **Benign Overfitting.** In the presence of noise in the data generating process (i.e. $\sigma_\epsilon^2 \neq 0$ in (1.2.1)), trained neural networks that exactly fit the training data still often can reliable predictions on unseen inputs [ZBH+17]. This seems to fly in the face of the traditional imperative to regularize models and is known as benign overfitting, a term coined in [BLLT20]. Explaining why and when it occurs is an important open problem.

A general language for explaining implicit bias remains elusive. However, there is a clear high-level intuition that might reasonably explain a part of why learned interpolants are often well-behaved. The basic idea is that in practice optimization is only possible with well-tuned architecture-dependent initialization schemes. The resulting distribution over functions $z^{(L+1)}(x;\theta_0)$ at the start of training (where θ_0 is chosen at random) specifies the initial conditions for optimization, and good optimization schemes used in practice are biased towards simple functions. Moreover, first order optimization methods are a kind of greedy local search. Hence, at least intuitively, neural network optimization will seek out a way to fit the training data that is "as close as possible" to the simple functions represented by randomly initialized networks.

Perhaps the most canonical example of this is regression with overparameterized linear

models in which optimization is performed by GD or SGD[2]. More precisely, we consider a linear model

$$z(x;\theta) = \sum_{j=1}^{P} \theta_j \phi_j(x) \tag{1.3.5}$$

with fixed basis functions $\phi_j : \mathbb{R}^{n_0} \to \mathbb{R}$ and a collection of training datapoints

$$\mathcal{D} = \{(x_\alpha, y_\alpha),\, \alpha = 1, \dots, |\mathcal{D}|\} \subseteq \mathbb{R}^{N_0+1}.$$

We will assume that the model is overparameterized, which in this case means

$$P = \#\text{ parameters } > \#\text{ datapoints } = |\mathcal{D}|\,.$$

Consider the empirical loss

$$\mathcal{L}_{\mathcal{D}}(\theta) = \sum_{(x_\alpha, y_\alpha)\in\mathcal{D}} \ell\left(z(x_\alpha;\theta), y_\alpha\right),$$

where for each y the function $z \mapsto \ell(z,y)$ is strictly convex and achieves a minimum only at on the diagonal $z = y$ (e.g. $\ell(z,y) = (z-y)^2$). Minimizing $\mathcal{L}_{\mathcal{D}}$ is equivalent to solving an underdetermined systems of linear equations with $|\mathcal{D}|$ equations and P unknowns. The set of solutions to such a system of equations is a hyperplane that is parallel to the kernel of the transpose of the $|\mathcal{D}| \times P$ feature matrix Φ defined by

$$\Phi := (\phi_1(\mathcal{D}), \dots, \phi_P(\mathcal{D}))\,, \qquad \phi_j(\mathcal{D}) = \left(\phi_j(x_1) \cdots \phi_j(x_{|\mathcal{D}|})\right)^T.$$

See Figure **??**. This hyperplane is the manifestation in this context of the infinitely many minima one generically expects for an empirical loss in an overparameterized neural network. Since the loss $\mathcal{L}$ is convex, any reasonable first order method such as gradient descent with a sufficiently small step size will converge to a global minimum.

The question of implicit bias is now: what determines which particular minimizer one will obtain? In this case, the answer comes by noting that the gradient of the loss

$$\nabla_\theta \mathcal{L}_{\mathcal{D}}(\theta) = \sum_{\alpha=1}^{|\mathcal{D}|} \partial_z \ell(z,y)\big|_{z=z(x_\alpha;\theta), y=y_\alpha} \cdot \Phi(x_\alpha) \in \mathrm{col}(\Phi) \tag{1.3.6}$$

always belongs to the column space of Φ. The key point is that $\mathrm{col}(\Phi)$ is perpendicular to the kernel of Φ^T and hence we conclude that gradient descent starting from θ_0 converges to the orthogonal projection of θ_0 onto the space of minima of $\mathcal{L}$. Put another way, gradient descent starting at θ_0 converges to the nearest, in the ℓ_2 sense, minimum of $\mathcal{L}$. This observation can be further understood by considering the decomposition

$$\theta = \theta_{||} + \theta_\perp, \qquad \theta_\perp \in \ker(\Phi^T), \quad \theta_{||} \in \mathrm{col}(\Phi). \tag{1.3.7}$$

Since $\nabla_\theta \mathcal{L}_{\mathcal{D}}(\theta) \in \mathrm{col}(\Phi)$, we see that $\theta_\perp$ does not change in the course of gradient descent. In contrast, $\mathcal{L}_{\mathcal{D}}$ is strictly convex on $\mathrm{col}(\Phi)$ and hence $\theta_{||}$ converges to the unique minimum

[2]The derivation below is certainly an old idea, and the author does not know its exact intellectual provenance. The author first learned about it from discussion with Misha Belkin at the Simons Institute in 2019. See also §11.3.2 in [DHP21].

of $\mathcal{L}_{\mathcal{D}}$ inside col(Φ). This unique minimum is precisely the minimal ℓ_2-norm minimizer of $\mathcal{L}$. The appearance of the ℓ_2 norm on $\mathbb{R}^P$ comes because we compute gradients using the ℓ_2 inner product.

An interesting consequence of the preceding discussion is that gradient descent starting at $\theta_0 = 0$ converges to the minimal ℓ_2-norm minimizer of $\mathcal{L}$. This helps to explain why implicit bias is sometimes referred to as implicit regularization: while we did not seem to explicitly penalize the ℓ_2 norm of the parameter vector θ when minimizing $\mathcal{L}_{\mathcal{D}}$, our choice of optimization algorithm implicitly does this for us.

Of course when $z(x;\theta)$ is non-linear in θ the preceding discussion – in particular the crucial relationship (1.3.6) – no longer holds.

Bibliography on Implicit Bias

Many excellent articles have been written on the subjects of implicit bias and benign overfitting with neural networks. Below is a far from representative list:

- **Logistic Regression.** The implicit bias of gradient descent toward maximum margin optima for vanilla logistic regression with separable data was shown in [SHN$^+$18]. This was then extended to more general homogeneous neural network in [LL19, JT20].

- **Linear Networks/Matrix Factorization.** The implicit bias of the optimizer so-called diagonal linear networks was worked out in [WGL$^+$20]. See also [ACHL19]. Other interesting directions include understanding the implicit bias of gradient descent in linear convolutional networks [GWB$^+$18] and for matrix factorization [GLSS18].

1.3.3 Q3. Feature/Transfer Learning.

Perhaps the most mathematically well-understood form of learning relies on approximation by linear spaces of functions. Such linear models take the form

$$z(x;\theta) = \sum_{j\geq 1} \theta_j \phi_j(x),$$

where $(\phi_j(x),\, j \geq 1)$ is a given set of functions. The choice of a "good" basis – one in which the unknown function f has in some sense an efficient expansion – is key to providing theoretical guarantees for how well one can estimate the vector of coefficients θ based on observations from f. Note that, in contrast, the i^{th} component of the output of a neural network can be written as

$$z_{i;\alpha}^{(L+1)}(x;\theta) = \sum_{j=1}^{n_L} W_{ij}^{(L+1)} \phi_j(x_\alpha;\theta), \qquad \phi_j(x_\alpha;\theta) = \sigma\left(z_{j;\alpha}^{(L)}\right).$$

Hence, neural network training can be viewed as learning both basis function and the coefficients in the linear combination simultaneously. It is widely believed that the ability to learn data-adaptive basis functions (a.k.a. features) is a key component of the success of neural networks and many fascinating articles have attempted to understand this point.

Bibliography on Feature/Transfer Learning

Many excellent articles have been written on the subjects of feature learning in neural networks. Below is a far from representative list:

- **Staircase Property and Leap Complexity.** Breakthrough work [AAM22] and its followup [AAM23] characterized the implicit bias of GD for learning of sparse polynomials with one layer networks. This is related to the impactful works [AGJ21, BAGJ22] about SGD in high dimensions.
- **Single-index Models.** A range of articles show that it one layer networks can efficiently learn planted one-dimensional structure in the data generating process. These articles include [BBSS22, AKLS23].
- **Multi-index Models.** A number of excellent papers concern learning multi-index models by gradient descent with one layer networks. These include [BMZ23, MHPG$^+$22, CWPPS23, BBPV23].

2 Analysis of One Layer Networks

2.1 Introduction

In this section, we analyze optimization of a hidden layer network with output dimension 1

$$z^{(2)}(x;\theta) = \frac{1}{\gamma\sqrt{N_1}} \sum_{i=1}^{N_1} W_i^{(2)} \sigma\left(\frac{1}{\sqrt{N_0}} W_i^{(1)} \cdot x\right), \qquad W_i^{(1)} \in \mathbb{R}^{N_0},\, W_i^{(2)} \in \mathbb{R} \tag{2.1.1}$$

trained by gradient flow

$$\frac{d}{dt}\theta_t = -\eta \nabla_\theta \mathcal{L}_{\mathcal{D}}(\theta_t), \qquad \theta_t = \left(W_i^{(1)}(t), W_i^{(2)}(t)\right)_{i=1}^{N_1} \tag{2.1.2}$$

to minimize the empirical mean squared error

$$\mathcal{L}_{\mathcal{D}}(\theta) := \frac{1}{|\mathcal{D}|} \sum_{(x,y)\in\mathcal{D}} \left(z^{(2)}(x;\theta) - y\right)^2 \tag{2.1.3}$$

over a dataset $\mathcal{D}$[1]. While our analysis focuses on gradient flow, it applies equally well to gradient descent with a sufficiently small step size. The initial condition θ_0 for optimization will be determined by setting

$$W_{ij}^{(1)}, W_i^{(2)} \sim \mathcal{N}(0,1) \text{ iid} \quad i = 1,\dots,N_1,\, j = 1,\dots,N_0. \tag{2.1.4}$$

We will describe optimization in the regime where the input dimension N_0, the output dimension N_2 and dataset size $|\mathcal{D}|$ are fixed but the hidden layer width N_1 tends to infinity. As we shall see, the size of the constant γ, which we have used to scale the neural network output in (2.1.1), will play a crucial role in our analysis:

- $\gamma = \Theta(1)$: This regime is known as the kernel or lazy or NTK regime. In this setting we will see in §2.2 that when $N_1 \to \infty$, neural networks are equivalent to linear models. More precisely, the entire trajectory of optimization converges to that of the linear model obtained by linearizing the network (2.1.1) around the initialization (2.1.4):

$$z^{(2)}(x;\theta) \quad \mapsto \quad z^{(2)}(x;\theta_0) + \left\langle \nabla_\theta z^{(2)}(x;\theta_0), \theta - \theta_0 \right\rangle. \tag{2.1.5}$$

  This somewhat surprising phenomenon was first discovered in the special case of one layer ReLU network in [DLT+18] and then derived for network at any fixed depth and a broad class of activations in [JGH18]. See also [ACGH19, DZPS19]. In §2.2 below, we will mainly follow the presentation in [LGJ20].

[1] The pre-factor η in (2.1.2) is a way to rescale the time variable t, which will be necessary in the mean-field analysis presented in §2.3.

- $\gamma = \Theta(\sqrt{N_1})$: This regime is known as the mean field or rich regime. We explain §2.3.2 that optimization can be equivalently re-formulated as either ℓ_2-based optimal transport or as a particularly simple McKean-Vlasov process. In this regime, neural networks are not linear models and learn complex data-dependent features, even when $N_1 \to \infty$. The exact nature of this feature-learning has been extensively studied (see e.g. [AAM22, AAM23, BMZ23, BBSS22, BBPV23]), but is far from completely understood.

  The optimal transport point of view was first discovered in two more or less concurrent articles [MMN18, CB18b]. An analysis closer to the interacting particle McKean-Vlasov point of view was first proposed in [RVE18], which was concurrent with the optimal transport papers, and then developed from a somewhat different point of view in articles such as [SS20, SS21, PN21].

  Unlike the NTK regime, it is usually unclear how to extend the mean-field analysis of wide neural networks to the setting of more than one hidden layer (see [**?**, **?**, **?**, **?**] for some notable exceptions). This remains an active and important area of research.

We sketch in §2.2 and §2.3.2 below the analysis of optimization in the kernel and mean-field regimes, respectively. Before doing so, we pause to understand the distribution of network output $x \mapsto z^{(2)}(x;\theta)$ and gradients $x \mapsto \nabla_\theta z^{(2)}(x;\theta)$ at initialization (i.e. when $\theta = \theta_0$) in §2.1.1. These properties will be used to study the mean field and NTK regimes.

2.1.1 One Layer Networks at Initialization

Distribution of Network Outputs

Let us agree for any network input $x \in \mathbb{R}^{N_0}$ to write

$$z^{(1)}(x) := \left(z_i^{(1)}(x),\, i = 1, \dots, N_1\right), \quad z_i^{(1)}(x) := \frac{1}{\sqrt{N_0}} W_i^{(1)} \cdot x \tag{2.1.6}$$

for *pre-activations in the first layer*. With this notation

$$z^{(2)}(x;\theta) = \frac{1}{\gamma\sqrt{N_0}} \sum_{i=1}^{N_1} W_i^{(2)} \sigma\left(z_i^{(1)}(x)\right).$$

As a starting point for understanding optimization, let us assume that the network weights are initialized at random as in (2.1.4) and determine the distribution of both the hidden layer pre-activations $z_i^{(1)}(x)$ and the network output $z^{(2)}(x)$ (see Chapter 3 for a treatment of the more complicated case of deeper networks). To start, recall that at initialization (2.1.4), the weight vectors $W_i^{(1)}$ are iid standard Gaussians in $\mathbb{R}^{N_0}$. The map $x \in \mathbb{R}^{N_0} \mapsto z^{(1)}(x) \in \mathbb{R}^{N_1}$ is thus a centered Gaussian field with iid entries

$$\mathrm{Cov}\left(z_i^{(1)}(x),\, z_j^{(1)}(x')\right) = \delta_{ij} K^{(0)}(x, x'), \qquad K^{(0)}(x, x') := \frac{x \cdot x'}{N_0}. \tag{2.1.7}$$

Further, since all weights weights are iid Gaussian, conditional on the first layer weights $W_i^{(1)}$, we find that the field $x \mapsto z^{(2)}(x;\theta_0)$ is a centered Gaussian field with conditional covariance

$$\mathrm{Cov}\left(z^{(2)}(x;\theta_0), z^{(2)}(x';\theta_0) \mid W_i^{(1)},\, 1 \le i \le N_1\right) = \frac{1}{\gamma^2 N_1} \sum_{i=1}^{N_1} \sigma\left(z_i^{(1)}(x)\right) \sigma\left(z_i^{(1)}(x')\right).$$

At finite width the full distribution of $x \mapsto z^{(2)}(x;\theta)$ is that of a Gaussian field with an independent random covariance and hence is not Gaussian. However, note from (2.1.7) that the summands in this random covariance are iid. Thus, we find in the infinite width limit $N_1 \to \infty$ that this covariance becomes deterministic:

$$K^{(2)}(x,x') := \lim_{N_1\to\infty} \frac{1}{\gamma^2 N_1} \sum_{i=1}^{N_1} \sigma\left(z_i^{(1)}(x)\right)\sigma\left(z_i^{(1)}(x')\right) = \frac{1}{\gamma^2}\mathbb{E}\left[\sigma\left(z_1^{(1)}(x)\right)\sigma\left(z_1^{(1)}(x')\right)\right],$$

where the expectation is with respect to the standard Gaussian $W_1^{(1)}$. If γ is a fixed positive number, then as $N_1 \to \infty$, the distribution over outputs of a randomly initialized one layer network is therefore a Gaussian field at infinite width. This simplification was first pointed out by Neal in [Nea96]. It was then extended to networks of any depth under various assumptions and levels of rigor in [LBN+18, Yai20, Han23, GARA18, HBSDN20, Yan19].

Distribution of Network Gradients

Here we discuss the distribution of the field of parameter-gradients $x \mapsto \nabla_\theta z^{(2)}(x;\theta)$ at initialization when N_1 grows. The dimension of the output of this field is $N_1(N_0+1)$, which diverges in this regime. However, as we'll see in §2.2, we need only understand the inner products of these gradient vectors (times the learning rate), i.e. the so-called *neural tangent kernel* [JGH18, ADH+19b],

$$\mathrm{NTK}(x,x';\theta) := \eta\left\langle \nabla_\theta z(x;\theta), \nabla_\theta z(x';\theta)\right\rangle \tag{2.1.8}$$

to understand many aspects of optimization. Explicitly, we have

$$\mathrm{NTK}(x,x';\theta) := \frac{\eta}{\gamma^2 N_1}\sum_{i=1}^{N_1} \sigma\left(z_i^{(1)}(x)\right)\sigma\left(z_i^{(1)}(x')\right) + \frac{x\cdot x'}{N_0}\sigma'\left(z_i^{(1)}(x)\right)\sigma'\left(z_i^{(1)}(x')\right). \tag{2.1.9}$$

Since by (2.1.7) the pre-activation fields $x \mapsto z_i^{(1)}(x)$ are iid, we find that at large N_1

$$\mathrm{NTK}(x,x';\theta) \approx \frac{\eta}{\gamma^2}\mathbb{E}\left[\sigma\left(z_i^{(1)}(x)\right)\sigma\left(z_i^{(1)}(x')\right) + \frac{x\cdot x'}{N_0}\sigma'\left(z_i^{(1)}(x)\right)\sigma'\left(z_i^{(1)}(x')\right)\right], \tag{2.1.10}$$

where the expectation is over the Gaussian distribution of the first layer weights. Assuming that our data is normalized so that $||x||^2 = \Theta(N_0)$, we find that this expectation is order 1 if and only if

$$\eta = \Theta\left(\gamma^2\right). \tag{2.1.11}$$

We will see below that this is indeed the correct scaling of η so that gradient flow (2.1.2) has a well-defined limit as $N_1 \to \infty$.

2.2 Analysis of Optimization in the Kernel Regime

In this section, we present an influential line of work on the so-called NTK (aka linear, kernel, or lazy) regime of neural networks [DLT+18, JGH18, LZB22a]. This regime occurs when we set the γ appearing the definition (2.1.1) to be a constant, which we shall take to be 1, independent of the network width N_1. The main result is the following

Theorem 2.2.1 ([DLT+18, JGH18, LZB22a, CB18a]). *Fix $L, n_0 \geq 1$ and $n_{L+1} = 1$ as well as $\sigma : \mathbb{R} \to \mathbb{R}$ with $\sigma \in C^2$ polynomially bounded and has infinitely many non-zero terms in its Hermite expansion. Define*

$$\mathcal{L}(\theta) = \sum_{i=1}^{m} \left(z^{(L+1)}(x_i; \theta) - y_i \right)^2, \quad (x_i, y_i) \sim \mathbb{P} \text{ iid.}$$

Consider

$$\frac{d}{dt}\theta_t = -\nabla_\theta \mathcal{L}(\theta_t)$$

with θ_0 as in (2.1.4). If $n_1, \ldots, n_L = \text{poly}(m)$ then under mild assumptions on $\mathbb{P}$ and with with high probability over the distribution of the training data and over initialization

(i) Optimization is successful in the sense that

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

(ii) The entire trajectory of optimization is close to its linearization around $t = 0$ in the sense that there exist $\alpha > 0$ so that

$$\sup_{t \geq 0} \left|\left| \theta_t - \theta_t^{\text{lin}} \right|\right| = O(\min\{n_1, \ldots, n_L\}^{-\alpha}) \tag{2.2.1}$$

where

$$\begin{aligned} \frac{d}{dt}\theta_t^{\text{lin}} &= -\nabla_\theta \mathcal{L}^{\text{lin}}(\theta_t^{\text{lin}}) \\ \mathcal{L}^{\text{lin}}(\theta) &= \sum_{i=1}^{m} \left(z^{(L+1),\text{lin}}(x_i; \theta) = y_i \right)^2 \\ z^{(L+1),\text{lin}}(x; \theta) &= z^{(L+1)}(x; \theta_0) + \nabla_\theta z^{(L+1)}(x; \theta_0)\,(\theta - \theta_0) \end{aligned}$$

For the estimates (2.2.1) we refer the reader to Appendix B of [BMR21]. The basic idea of the proof of Theorem 2.2.1 is the same for any network depth L and relies on two key ideas:

- **Idea 1.** Rewriting the training dynamics in terms of the neural tangent kernel, defined in (2.1.8).
- **Idea 2.** Checking that the change in the neural tangent kernel over the course of training tends to zero in the infinite width limit.

We illustrate both these ideas for $L = 1$ in §2.2.1 and §2.2.1 below.

2.2.1 Role of the NTK in Optimization

Let us consider for a moment a more general optimization problem of fitting the parameters of a generic differentiable model $z(x; \theta)$ by gradient flow

$$\frac{d}{dt}\theta_t = -\eta \nabla_\theta \mathcal{L}(\theta_t) \tag{2.2.2}$$

on the empirical loss

$$\mathcal{L}(\theta) = \frac{1}{2}\sum_{i=1}^{m} \left(z(x_i;\theta) - y_i\right)^2$$

over a dataset $\mathcal{D} = \{(x_i, y_i)\}$. As a function of θ, the loss $\mathcal{L}(\theta)$ is rather complicated. However, $\mathcal{L}$ is a simple function of the values

$$z(\mathcal{D};\theta) = (z(x_i;\theta),\, (x_i, y_i) \in \mathcal{D})$$

the model takes on the training data. A simple computation shows that

$$\frac{d}{dt} z(\mathcal{D};\theta_t) = -\mathrm{TK}(\theta_t)\left(z(\mathcal{D};\theta_t) - Y\right),$$

where

$$\mathrm{TK}(\theta) := \left(\eta \left\langle \nabla_\theta z(x_i;\theta), \nabla_\theta z(x_j;\theta)\right\rangle\right), \quad 1 \le i, j \le |\mathcal{D}|), \qquad Y = (y_1, \ldots, y_m),$$

where $\mathrm{TK}(\theta)$ is sometimes called the tangent kernel of the model [LZB22a]. Geometrically this means that $\mathrm{TK}(\theta_t)^{-1}$ plays the role of a Riemannian metric on the tangent space at the function $z(\cdot;\theta_t)$ that is used to perform gradient descent in the new coordinates $z(\mathcal{D};\theta)$. This implies the following simple but fundamental result

Lemma 2.2.2. *Suppose there exists $\delta > 0$ so that we have the following inequality of PSD matrices*

$$\mathrm{TK}(\theta_t) \ge \delta I \qquad \forall t \ge 0. \tag{2.2.3}$$

Suppose further that

$$\mathcal{L}(\theta) = \sum_{i=1}^{m} \ell(z(x_i;\theta), y_i)$$

with $\ell(a, b)$ a jointly strictly convex function in a, b. Then for η sufficiently small

$$\mathcal{L}(\theta_t) - \min_\theta \mathcal{L}(\theta) \le e^{-c\eta t}\left(\mathcal{L}(\theta_0) - \min_\theta \mathcal{L}(\theta)\right)$$

for some $c > 0$.

Proof. The condition (2.2.3) ensures that

$$\left\langle \mathrm{TK}(\theta_t)\nabla_z \mathcal{L}(z(\mathcal{D})), \nabla_z \mathcal{L}(z(\mathcal{D};\theta_t))\right\rangle \ge c\,||\nabla_z \mathcal{L}(z(\mathcal{D};\theta_t))||^2.$$

Since $\mathcal{L}$ is strictly convex as a function of the predictions of the model $z(\cdot;\theta)$ on the training data, this guarantees exponential convergence to a minimum. □

Corollary 2.2.3. *Suppose*

$$\mathcal{L}(\theta) = \sum_{i=1}^{m} \left(z(x_i;\theta) - y_i\right)^2.$$

Then, if $\Theta(\theta_t) \ge \delta I$ for all $t \ge 0$, then optimization is successful

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

Linearization at Large Width

To apply Lemma 2.2.2 in our setting of one hidden layer networks we must check that the condition (2.2.3) holds with high probability. This is typically done in two steps

(i) Show that (2.2.3) holds at initialization, i.e. that

$$\mathrm{NTK}(\theta_0) \geq \delta I, \qquad \delta > 0. \tag{2.2.4}$$

(ii) Show that the NTK remains positive definite during training:

$$\sup_{t \geq 0} ||\Theta(\theta_0) - \Theta(\theta_t)|| \leq \frac{\delta}{2}. \tag{2.2.5}$$

Let's see how this works in our particularly simple case of one layer networks with NTK parameterization in which $\gamma, \eta = \Theta(1)$. The key observation for step (i) is that by (2.1.10) the matrix $\mathrm{NTK}(\theta_0)$ is actually an average of N_1 iid random matrices. Hence, when N_1 is large it concentrates rapidly around it's mean, which has the form

$$\mathbb{E}\left[\mathrm{NTK}(\theta_0)\right] = \left(\mathbb{E}\left[\sigma\left(z_i^{(1)}(x_i)\right)\sigma\left(z_i^{(1)}(x_j)\right) + \frac{x \cdot x'}{N_0}\sigma'\left(z_i^{(1)}(x_i)\right)\sigma'\left(z_i^{(1)}(x_j)\right)\right]\right)_{1 \leq i,j \leq |\mathcal{D}|}. \tag{2.2.6}$$

It therefore remains to check that this matrix is strictly positive definite with high probability over the dataset $\mathcal{D}$. The mean of the NTK is precisely the Gram matrix for the vectors

$$\left(\sigma\left(\frac{W \cdot x}{\sqrt{N_0}}\right), \frac{x_1}{\sqrt{N_0}}\sigma'\left(\frac{W \cdot x}{\sqrt{N_0}}\right), \ldots, \frac{x_{N_0}}{\sqrt{N_0}}\sigma'\left(\frac{W \cdot x}{\sqrt{N_0}}\right)\right), \qquad (x, y) \in \mathcal{D},$$

viewed as elements of $L^2(\mathbb{R}^{N_0}, d\mu)$, where μ is the standard Gaussian measure on W. Thus, showing that $\mathbb{E}\left[\mathrm{NTK}(\theta_0)\right]$ is positive definite is equivalent to checking that with high probability over $\mathcal{D}$, these vectors are linearly independent in this Hilbert space. This requires a bit of work but is generally an elementary exercise given some genericity conditions on the distribution of $(x, y) \in \mathcal{D}$ and the non-linearity σ (i.e. σ is not a polynomial). For instance, the interested reader can look at Appendix A in [DLT+18] and Theorem 5.2 in [OS20]. Finally to check step (ii), we begin with the following observation:

$$||\mathrm{NTK}(\theta_0) - \mathrm{NTK}(\theta_t)|| \leq \int_0^t \max_{(x,y) \in \mathcal{D}} ||\mathrm{Hess}_\theta z(x; \theta_s)||\, ||\nabla \mathcal{L}(\theta_s)||\, ds,$$

where $\mathrm{Hess}_\theta z(x; \theta)$ is the Hessian with respect to the parameters of the neural network. The estimate (2.2.5) then reduces to deriving an inequality of the form

$$||\mathrm{Hess}_\theta z(\theta)|| \leq \frac{\mathrm{poly}\left(\#\text{training datapoints}\right)}{\sqrt{\text{width}}}, \qquad \forall\, ||\theta - \theta_0|| \leq R.$$

To see why this must be true, note that

$$\frac{\partial^2}{\partial W_i^{(\ell_1)} \partial W_j^{(\ell_2)}} z^{(2)}(x; \theta) = \frac{1}{\sqrt{N_1}} \delta_{ij} \cdot \Theta\left(\text{order 1 object}\right), \qquad \ell_1, \ell_2 \in \{1, 2\}. \tag{2.2.7}$$

Hence,

$$\mathrm{Hess}_\theta z^{(2)}(x;\theta) = \begin{pmatrix} H_1(\theta) & 0 & \cdots & 0 \\ 0 & H_2(\theta) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & H_{n_1}(\theta) \end{pmatrix}, \qquad H_i(\theta) = \mathrm{Hess}_{W_i^{(1)}, W_i^{(2)}} z^{(2)}(x;\theta)$$

The pre-factor in (2.2.7) shows that

$$\left|\left|\mathrm{Hess} z^{(2)}(x;\theta_0)\right|\right| = O\left(\frac{1}{\sqrt{N_1}}\right).$$

So at initialization a one layer network is close to it's linearization. Moreover, some simple perturbation theory shows that

$$||\theta - \theta_0|| \text{ not too large} \qquad \Longrightarrow \qquad \left|\left|\mathrm{Hess}_\theta z^{(2)}(x;\theta)\right|\right| = O\left(\frac{1}{\sqrt{N_1}}\right),$$

with the implicit constant depending on the training dataset $\mathcal{D}$. Completing proof then requires showing that the parameters don't leave a sufficiently large ball around initialization.

2.3 One Layer Networks in the Mean Field Regime

In the mean-field regime, a one layer neural network takes the form

$$z^{(2)}(x;\theta) = \frac{1}{N_1}\sum_{i=1}^{N_1} W_i^{(2)} \sigma\left(z_i^{(1)}(x)\right), \qquad z_i^{(1)}(x) = \frac{W_i^{(1)} \cdot x}{\sqrt{N_0}}. \tag{2.3.1}$$

This corresponds to setting $\gamma = \sqrt{N_1}$ in (2.1.6). While the choice of γ may only appear to be a matter of parameterization, we will see in §2.3.2 it has a significant effect on the nature of first order optimization with one layer networks. Before explaining this, we take a brief but useful detour in §2.3.1 to explain how to view any one layer neural network in the mean field regime as a probability measure.

2.3.1 One Layer Networks as (Empirical) Measures

The neural network (2.3.1) can naturally be rewritten in terms of a measure on $\mathbb{R}^{N_0+1}$:

$$z^{(2)}(x;\theta) = \int_{\mathbb{R}^{N_0+1}} W^{(2)} \sigma\left(\frac{W^{(1)} \cdot x}{\sqrt{N_0}}\right) d\rho_\theta(W^{(1)}, W^{(2)}),$$

where

$$\rho_\theta(W^{(1)}, W^{(2)}) := \frac{1}{N_1}\sum_{i=1}^{N_1} \delta_{\left\{W_i^{(1)}, W_i^{(2)}\right\}}$$

is simply the empirical measure of incoming and outgoing weights across all neurons. Thus, there is map from the space of one layer networks to the space of probability measures $\mathcal{P}(\mathbb{R}^{N_0+1})$ on $\mathbb{R}^{N_0+1}$. Except on a measure 0, the image of the networks with hidden layer width N_1 is the collection of empirical measures with N_1 atoms in $\mathcal{P}(\mathbb{R}^{N_0+1})$. Any question

about one hidden layer networks can, using the mapping from one layer networks to probability measures, be translated into a question about the linear space $\mathcal{P}(\mathbb{R}^{N_0+1})$. This is often useful since one can now reason about networks of all widths $N_1 \geq 1$ in a uniform way.

Consider for example, the classical question of which functions $f : \mathbb{R}^{N_0} \to \mathbb{R}$ can be well-approximated by one hidden layer networks with non-linearity σ and width N_1? The identification between one layer networks and measures naturally decomposes this into two questions:

1. **Approximation at infinite width**. Which functions $f : \mathbb{R}^{N_0} \to \mathbb{R}$ can be written
$$f(x) = \int_{\mathbb{R}^{N_0+1}} W^{(2)}\sigma(W^{(1)} \cdot x)d\mu(W^{(1)}, W^{(2)}) \tag{2.3.2}$$
for some $\mu \in \mathcal{P}(\mathbb{R}^{n_0+1})$?
2. **Rate of approximation**. Given a measure $\mu \in \mathcal{P}(\mathbb{R}^{n_0+1})$, how well can it be approximated by an empirical measure with N_1 atoms?

The answer to question 1 is that for any "reasonable" non-linearity σ (e.g. not a polynomial on some interval), every "reasonable" function (say continuous) can be written in the form (2.3.2). This is essentially the point of view taken by Hornik [HSW89]. Obtaining sharp answers to the second question is more subtle (see [Bar92, MP16]). One simple approach is to note that every $\mu \in \mathcal{P}(\mathbb{R}^{N_0+1})$ can be approximated in the infinite width limit by such measures by the law of large numbers
$$\frac{1}{N_1}\sum_{i=1}^{N_1} \delta_{W_i^{(1)}, W_i^{(2)}} \quad \to \quad \mu, \qquad \left(W_i^{(1)}, W_i^{(2)}\right) \sim \mu \text{ i.i.d.}, \tag{2.3.3}$$
where the convergence is in the weak sense and occurs almost surely. For example using a uniform central limit theorem to quantify the rate of convergence in (2.3.3) one may obtain as in Barron [Bar92] a $N_1^{-1/2}$ upper bound on the rate of convergence for certain function spaces.

2.3.2 Optimization of One Layer Networks in the Mean Field Regime

The identification between one layer networks and empirical measures has been used by a variety of authors to understand not just approximation with one layer networks but also optimization [MMN18, CB18b, RVE18, SS20, SS21, BP22]. The high-level takeaways from these articles are as follows:

- **Optimal transport.** Suppose that our training data is generated by
$$y = f(x),$$
for some fixed $f : \mathbb{R}^{N_0} \to \mathbb{R}$, which admits a representation
$$f(x) = \int_{\mathbb{R}^{N_0+1}} W^{(2)}\sigma\left(W^{(1)}x\right) d\mu_f(W^{(1)}, W^{(2)}).$$
Suppose further the empirical loss of the training dataset is replaced by the population loss
$$\mathcal{L}(\theta) = \mathbb{E}_{(x,y)}\left[\left(z^{(2)}(x;\theta) - y\right)^2\right].$$

The trajectory of the empirical measures

$$\rho_t := \rho_{\theta_t} = \frac{1}{N_1}\sum_{i=1}^{N_1} \delta_{W_i^{(1)}(t), W_i^{(2)}(t)}, \tag{2.3.4}$$

is that of ℓ_2-based optimal transport. The initial condition ρ_0 is determined by the distribution of weights at initialization and the final condition is μ_f. This point of view was developed in [MMN18], which describes also many extensions, such as how gradient flow on an empirical loss approximates gradient on the population loss.

- **Particles with Mean-Field Interactions.** In the notation of (2.1.2), define

$$X_i(t) := \left(\frac{1}{\sqrt{N_0}} W_i^{(1)}(t) \cdot x_\alpha, W_i^{(2)} \right)_{\alpha=1}^{|\mathcal{D}|}, \qquad i = 1, \dots, N_1$$

and write

$$\widehat{\rho}_t := \frac{1}{N_1}\sum_{i=1}^{N_1} \delta_{X_i(t)}.$$

A direct computation shows that, even at finite width, the dynamics of $X_i(t)$ are form a closed system of mean-field type in the sense that the time-derivative of the "state" of each "particle" $X_i(t)$ depends on the other particles only through their empirical measure, i.e.

$$\frac{d}{dt} X_i(t) = F\left(X_i(t), \widehat{\rho}_t\right), \tag{2.3.5}$$

for an explicit function F (see (2.3.7), (2.3.8)). In this way, the dynamics at any finite width of a one hidden layer network trained by gradient flow over a fixed dataset is the discretization of the Vlasov equation with random initial conditions. This is, with some variations, the point of view taken in [BP22, SS20, SS21].

We will not attempt to reproduce all the technical details of the mean field analysis of one layer networks. Instead, we explain the basic idea, following the ideas developed in [BP22]. That is we consider optimizing the parameters of a mean-field neural network (2.3.1) by gradient flow

$$\frac{d}{dt}\theta_t = -N_1 \nabla_\theta \mathcal{L}(\theta), \qquad \mathcal{L}(\theta) = \frac{1}{2}\sum_{(x_\alpha, y_\alpha)\in\mathcal{D}} \left(z^{(2)}(x_\alpha; \theta) - y_\alpha \right)^2 \tag{2.3.6}$$

on the empirical MSE of a fixed training dataset, where the N_1 scaling of the learning rate is dictated comes from the relation (2.1.11). A core observation in [BP22] is that the dynamics (2.3.6) can be exactly re-written as following

$$\frac{d}{dt} z_i^{(1)}(x_\beta; \theta_t) = \frac{1}{|\mathcal{D}|}\sum_{(x_\alpha, y_\alpha)\in\mathcal{D}} \Delta_{\alpha t} W_i^{(2)}(t) \sigma'\left(z_{i;\alpha}^{(1)}(t) \right) \frac{x_\alpha \cdot x_\beta}{N_0} \tag{2.3.7}$$

$$\frac{d}{dt} W_i^{(2)}(t) = \frac{1}{|\mathcal{D}|}\sum_{(x_\alpha, y_\alpha)\in\mathcal{D}} \Delta_{\alpha t} \sigma\left(z_{i;\alpha}^{(1)}(t) \right), \tag{2.3.8}$$

where the residuals $\Delta_{\alpha t}$ are functions only of the empirical measures $\widehat{\rho}_t$:

$$\Delta_{\alpha t} = y_\alpha - z_\alpha^{(2)}(t) = y_\alpha - \frac{1}{N_1}\sum_{i=1}^{N_1} W_i^{(2)}\sigma\left(z_{i;\alpha}^{(1)}(t)\right) = y_\alpha - \left\langle \widehat{\rho}_t, W^{(2)}\sigma\left(z_\alpha^{(1)}\right)\right\rangle.$$

The equations (2.3.7) and (2.3.8) are precisely of the form (2.3.5). Moreover, the right hand side F is Lipschitz when σ' is Lipschitz. In this case, by propagation of chaos [**?**, **?**], we find as $N_1 \to \infty$ that $\widehat{\rho}_t$ obeys the following deterministic evolution:

$$\partial_t \widehat{\rho}_t = -\mathrm{div}\left(F\left(x, \widehat{\rho}_t\right) \cdot \widehat{\rho}_t\right).$$

Rewriting this slightly to obtain evolution equations for the empirical measures ρ_t from (2.3.4), we obtain the usual continuity equation formulation of optimal transport (see Remark 1.5 in [SS20] for details).

2.4 Difference Between the Mean Field and NTK Regimes

To conclude this chapter, we investigate why the seemingly small difference in parameterization – changing γ from 1 to $\sqrt{N_1}$ – led to such a big difference between the mean field and NTK regimes. To understand this we consider the change in the pre-activations. At initialization in the NTK parameterization we have

$$\frac{d}{dt} z_{i;\beta}^{(1)}(t) = \frac{1}{|\mathcal{D}|}\sum_{(x_\alpha, y_\alpha)\in\mathcal{D}} \Delta_{\alpha 0}\frac{1}{\sqrt{N_1}} W_i^{(2)}(0)\sigma'\left(z_{i;\alpha}^{(1)}(0)\right)\frac{x_\alpha \cdot x_\beta}{N_0}.$$

Computing the mean and variance with respect to the random initialization of $W_i^{(1)}(0)$ and $W_i^{(2)}(0)$ we find

$$\frac{d}{dt} z_{i;\beta}^{(1)}(t) \text{ is typically order } \frac{1}{\sqrt{N_1}}.$$

In contrast,

$$\left.\frac{d}{dt}\right|_{t=0} z_\beta^{(2)}(t) \text{ is typically order 1.}$$

Thus, in the NTK regime, the network output moves much faster than the pre-activations in the hidden layer. In a regression task (where the network output only needs to move a finite amount for every datapoint to minimize the loss) we thus expect that in the course of optimization the change in each hidden layer pre-activation has order $1/\sqrt{N_1}$. Since there are N_1 such pre-activations, this does not mean that in the NTK regime we can neglect the change in the hidden layer weights. Instead, we expect that we can Taylor expand the change in the network predictions with respect to the change in the hidden layer pre-activations and keep only the first order contribution. This precisely corresponds to linearizing the model around initialization and is the content of Theorem 2.2.1.

Bibliography on the Analysis of One Layer Networks

- mean field
  - [BP22] + [BNL$^+$23]
  - [MMN18]
  - [SS20, SS21] + others
  - [RVE18] + others?
  - [CB18b] + others?
  - [Yan19, YH21]
- NTK
  - [JGH18]
  - [**?**]
  - [SJL18]
  - [DLT$^+$18, DZPS19]
  - [LZB22a, LZB22b]
  - [HN20a, RYH22]
- Other ways to leave NTK regime:
  - Large P [AAM22, AAM23]
  - Large η [LBD$^+$20]
  - Time-scale separation [BMZ23]
- High-dimensional optimization:
  - Paquette +
  - Inbar +
  - Krzakala +
  - Ben Arous +
  - Bietti / Bruna +

3 Analysis of Deep Neural Networks at Init

In the previous chapter we studied learning with one hidden layer networks at large or infinite width. In the NTK regime we found that when trained by gradient descent on a fixed dataset, shallow networks are equivalent to linear models for the purposes of regression tasks (see §2.2). This kind of result extends readily to networks with any fixed depth [DZPS19, JGH18, LZB22a, ADH+19a]. We found, in contrast, that in the mean-field regime learning with one layer neural networks exhibits complex non-linear learning dynamics. Unlike the NTK-type analysis, most approaches to the mean-field approaches to neural networks have proved difficult to generalize beyond the case of a single hidden layer. A notable exception is the excellent article [BP22].

Our purpose in the next two chapters is to understand something about what happens when learning with networks in which depth and width are both large. The answer to this question is far from settled, and our purpose here is to survey a few related ideas. In the present Chapter we start by studying the behavior of neural networks at initialization, i.e. with random weights. Specifically:

- We begin in §3.1 by giving a precise definition of a random neural network.

- We proceed in §3.2 by investigating the relatively simple case of deep linear networks at initialization, which are simply products of random matrices. This is a fascinating subject in its own right. The main lesson we shall learn is that the ratio of the network's depth to width plays the role of a kind of "effective depth." In particular, we find that the limits as depth and width diverge do not commute.

- We then discuss in §3.3 that at any finite depth L random neural networks converge to centered Gaussian processes with a covariance kernel $K^{(L)}$ in the infinite width limit. In §3.3.3 we investigate the properties of $K^{(L)}$ at large depth L. Our core observation is that as long as the non-linearity is not identity, these kernels are degenerate at large depth L.

- In §3.4 study the distribution of the outputs of wide neural networks perturbatively with respect to the network width. We explain how the first order correction – captured by the fourth cumulant – scales like the ratio of network depth to width. In this way we find, much as in §3.2, that the depth to width ratio controls the distance to the Gaussian process limit.

- Finally, in §**??**, we introduce so-called *shaped* (i.e. weakly non-linear) activation functions, which are precisely those non-linearities for which the Gaussian process kernel $K^{(L)}$ has a non-degenerate large L limit.

3.1 What is a Random Neural Network?

Consider a random fully connected neural network with input dimension N_0, L hidden layers of widths $N_1, \dots, N_L$, output dimension N_{L+1}, and non-linearity σ. Recall from §1.1.1 that this means that an input $x \in \mathbb{R}^{N_0}$ produces an output $z^{(L+1)}(x) \in \mathbb{R}^{N_{L+1}}$ through a sequence of hidden layer representations $z^{(\ell)}(x)$

$$z_i^{(\ell+1)}(x) = \begin{cases} \frac{1}{\sqrt{N_\ell}} \sum_{j=1}^{N_\ell} W_{ij}^{(\ell+1)} \sigma\left(z_j^{(\ell)}(x)\right), & \ell \geq 1 \\ \frac{1}{\sqrt{N_0}} \sum_{j=1}^{N_0} W_{ij}^{(1)} x_j, & \ell = 0 \end{cases}.$$

A random neural network is obtain by taking the network weights to be random:

$$W_{ij}^{(\ell)} \sim \mathcal{N}(0, C_W) \text{ independent}, \quad 1 \leq \ell \leq L+1,\ 1 \leq j \leq N_{\ell-1},\ 1 \leq i \leq N_\ell.^1 \tag{3.1.1}$$

3.2 Case Study: Deep Linear Networks at Initialization

Before proceeding to the study of non-linear networks, we consider here the behavior at initialization of deep linear networks, which are obtained by taking $\sigma(t) = t$. In this case, the network output is obtained from the input through a product of matrices:

$$z^{(L)}(x) = \frac{1}{\sqrt{N_L}} W^{(L+1)} \cdots \frac{1}{\sqrt{N_0}} W^{(1)} x.$$

Such matrix products have been well-studied in two regimes:

- **Free Probability.** In this regime the network depth L is fixed and the matrix sizes N_ℓ tend to infinity. This setting is characterized by matrix spectra that are universal (i.e. independent of the exact distribution of matrix entries) and maximize an appropriate non-commutative notion of entropy [AGZ10, MS17].
- **Ergodic Theory.** In this regime the network widths are typically all equal $N_\ell \equiv N$ and held constant while the network depth L tends to infinity. This setting is characterized by rescaled matrix spectra (i.e. Lyapunov exponents) that are minimal entropy in the sense that they converge almost surely to deterministic limits that are not universal (i.e. depend on the details of distribution of matrix entries). See [FK60, B$^+$12].

To understand the interplay between depth and width in deep linear networks one must consider the deep-and-wide regime in which both N and L are large. The key point is that the limits as $N, L \to \infty$ don't commute and new phenomena arise. Our purpose here is to illustrate this in the simplest possible setting. For this, let us assume that $||x|| = 1$ and try to understand what is perhaps the simplest random variable associated to our random matrix product

$$X_{N,L+1} := \left|\left| z^{(L+1)}(x) \right|\right|,$$

which measures the magnitude of the network output at a fixed input. In order to understand its distribution recall that for any $k \geq 1$ a chi-squared random variable with k degrees of freedom is given by

$$\chi_k^2 :\stackrel{d}{=} \sum_{j=1}^{k} X_j^2, \qquad X_j \sim \mathcal{N}(0,1) \text{ } iid.$$

Note that for any unit vector u if $W \in \mathbb{R}^{N \times N'}$ is a matrix with $W_{ij} \sim \mathcal{N}(0,1)$ then

$$\frac{1}{\sqrt{N'}} W u \overset{d}{=} \mathcal{N}\left(0, \frac{1}{N'} \mathrm{I}_N\right) \qquad \Rightarrow \qquad ||Wu||^2 \overset{d}{=} \frac{1}{N'} \chi^2_N, \quad ||Wu|| \perp \frac{Wu}{||Wu||}, \tag{3.2.1}$$

where $X \perp Y$ means X is independent of Y. To use this let's write

$$\begin{aligned} X_{N,L+1} &= \left|\left| \frac{1}{N_L} W^{(L+1)} \cdots \frac{1}{\sqrt{N_0}} W^{(1)} x \right|\right| \\ &= \left|\left| \frac{1}{\sqrt{N_L}} W^{(L+1)} \cdots \frac{1}{\sqrt{N_1}} W^{(2)} \frac{W^{(1)} x}{||W^{(1)} x||} \right|\right| \left|\left| \frac{1}{\sqrt{N_0}} W^{(1)} x \right|\right|. \end{aligned} \tag{3.2.2}$$

This presentation of $X_{N,L}$ is actually a product of two independent terms due to (3.2.1)! Proceeding in this way, we obtain the following equality in distribution:

$$X_{N,L+1} \overset{d}{=} \frac{\sqrt{N_{L+1}}}{\sqrt{N_0}} ||x|| \exp\left[\sum_{\ell=1}^{L+1} Y_\ell\right], \qquad Y_\ell \sim \frac{1}{2} \log\left(\frac{1}{N_\ell} \chi^2_{N_\ell}\right) \text{ independent.} \tag{3.2.3}$$

A simple computation shows that

$$\mathbb{E}\left[\frac{1}{2} \log\left(\frac{1}{N} \chi^2_N\right)\right] = -\frac{1}{4N} + O(N^{-2}), \qquad \mathrm{Var}\left[\frac{1}{2} \log\left(\frac{1}{N} \chi^2_N\right)\right] = \frac{1}{4N} + O(N^{-2}).$$

Thus, we see that

$$X_{N,L+1} \overset{L \gg 1}{\approx} \frac{\sqrt{N_{L+1}}}{\sqrt{N_0}} ||x|| \exp\left[\mathcal{N}\left(-\frac{\beta}{4}, \frac{\beta}{4}\right)\right],$$

where

$$\beta := \frac{1}{N_1} + \cdots + \frac{1}{N_L} \simeq \frac{L}{N}$$

is essentially the depth-to-width ratio when all the hidden layer widths N_ℓ are proportional to a single large constant N. We see moreover that taking N_ℓ large in each layer tries to make each Y_ℓ close to 1 but with errors of size $1/N$. However, these orders add up in the exponent of (3.2.3) and give an L/N contribution.

Before returning to the study of non-linear networks we make one final remark. Namely, suppose we now consider a deep linear network with output dimension $N_{L+1} = 1$. Applying the same argument in (3.2.2) (but this time "backwards" starting with $x = W^{(L+1)}$), we find for any fixed L that

$$\lim_{N_1, \ldots, N_L \to \infty} \frac{1}{N_L} W^{(L+1)} \cdots \frac{1}{\sqrt{N_0}} W^{(1)} = \mathcal{N}\left(0, \frac{1}{N_0} I_{N_0}\right).$$

Hence, for any fixed value of L, the field $x \mapsto z^{(L+1)}(x)$ converges as $N_1, \ldots, N_L \to \infty$ to a centered Gaussian process with normalized Euclidean covariance

$$\lim_{N_1, \ldots, N_L \to \infty} \mathrm{Cov}\left(z^{(L+1)}(x), z^{(L+1)}(x')\right) = \frac{1}{N_0} \left\langle x, x' \right\rangle,$$

which is actually independent of L. We will see in the next section that this kind of Gaussian process result holds also for networks with general non-linear activations.

3.3 GP Limit at Infinite Width and Finite Depth

The starting point for our analysis of random neural networks with general activations is to understand their behavior at infinite width $N_1, \dots, N_L \to \infty$. To state the main result, let us agree that, given a $\mu : \mathbb{R}^{N_0} \to \mathbb{R}$ and a kernel $K : \mathbb{R}^{N_0} \times \mathbb{R}^{N_0} \to \mathbb{R}$, we will write $\mathcal{GP}(\mu, K)$ for the Gaussian process with mean μ and covariance K.

Theorem 3.3.1. *Fix L, σ, N_0, N_{L+1} and a non-linearity $\sigma : \mathbb{R} \to \mathbb{R}$ that is polynomially bounded in the sense that*

$$\exists k \geq 1 \text{ s.t. } \sup_{x \in \mathbb{R}} (1 + |x|^{-k}) \, |\sigma(x)| < \infty.$$

Then, as $N_1, \dots, N_L \to \infty$, the sequence of fields $x \mapsto z^{(L+1)}(x)$ converges in distribution to a mean zero Gaussian process with iid components:

$$\lim_{N_1, \dots, N_L \to \infty} z^{(L+1)}(\cdot) \overset{d}{=} \mathcal{GP}(0, K^{(L)})^{\otimes N_{L+1}},$$

where

$$\lim_{N_1, \dots, N_L \to \infty} \mathrm{Cov}\left(z_i^{(L+1)}(x_\alpha), z_j^{(L+1)}(x_\beta) \right) = \delta_{ij} K^{(L)}(x_\alpha, x_\beta),$$

with

$$K^{(\ell)}(x_\alpha, x_\beta) = \begin{cases} C_W \mathbb{E}_{K^{(\ell-1)}} \left[\sigma\left(z_1(x_\alpha)\right) \sigma\left(z_1(x_\beta)\right) \right], & \ell \geq 1 \\ \frac{1}{N_0} x_\alpha \cdot x_\beta, & \ell = 0 \end{cases} \tag{3.3.1}$$

and we've used the symbol $\mathbb{E}_K[\cdot]$ to denote the expectation over a distribution where $x \mapsto z_i^{(\ell)}(x)$ are centered Gaussian processes with covariance K that are independent for different i.

Remark 3.3.2. *The GP limit holds not only for fully connected architectures for also for virtually any feed-forward architecture. See e.g. [LBN+18, Yan19, HBSDN20]. See [Han23] for some universality results regarding the case of fully connected networks.*

We explain how to prove Theorem 3.3.1 in §3.3.2, taking for granted an important and much harder structural result (Theorem 3.3.3) presented in §3.3.1.

3.3.1 Structural Properties of Random Neural Networks at Large Width

The starting point for virtually every analysis of random neural network is the observation that the sequence $\left\{ z^{(\ell)}(x), \ell = 1, \dots, L+1 \right\}$ is a Markov chain. Moreover, the transition probabilities are Gaussian in the sense that, conditional on $z^{(\ell)}$, the fields $x \mapsto z_i^{(\ell+1)}(x)$ are iid mean zero Gaussian processes with conditional covariance

$$\begin{aligned} \widehat{K}^{(\ell)}(x_\alpha, x_\beta) :&= \mathrm{Cov}\left(z_m^{(\ell+1)}(x_\alpha), \, z_m^{(\ell+1)}(x_\beta) \mid \mathcal{F}^{(\ell)} \right) \\ &= C_b + \frac{C_W}{n_\ell} \sum_{j=1}^{n_\ell} \sigma\left(z_j^{(\ell)}(x_\alpha) \right) \sigma\left(z_j^{(\ell)}(x_\beta) \right). \end{aligned}$$

These conditional covariances are examples of *collective observables*

$$\mathcal{O}_f^{(\ell)} := \frac{1}{n_\ell} \sum_{i=1}^{n_\ell} f\left(z_i^{(\ell)}(x_\alpha),\ \alpha \in A\right).$$

The key technical result is the following:

Theorem 3.3.3 (Structure Theorem for Collective Observables)**.** *Suppose $\sigma : \mathbb{R} \to \mathbb{R}$ is polynomially bounded. For $k \geq 1$ suppose that $f_1, \ldots, f_k : \mathbb{R}^{|A|} \to \mathbb{R}$ are polynomially bounded. Then*

$$\mathbb{E}\left[\prod_{j=1}^{k}\left(\mathcal{O}_{f_j}^{(\ell)} - \mathbb{E}\left[\mathcal{O}_{f_j}^{(\ell)}\right]\right)\right] = O\left(N^{-\lceil \frac{k}{2} \rceil}\right), \tag{3.3.2}$$

where

$$N := \min\{N_1, \ldots, N_\ell\}.$$

Proof. See Theorem 3.1 and Lemma 7.5 in [Han22]. □

3.3.2 Derivation of GP Limit: Proof of Theorem 3.3.1 Modulo Theorem 3.3.3

We will show convergence of finite-dimensional distributions and leave tightness as an exercise. For this, let us fix a finite collection

$$x_A := (x_\alpha,\ \alpha \in A), \qquad x_\alpha \in \mathbb{R}^{N_0}$$

of $|A|$ distinct network inputs and agree to write

$$z_m^{(\ell)}(x_A) := \left(z_m^{(\ell)}(x_\alpha),\ \alpha \in A\right).$$

By Levy's continuity theorem we seek to show that for

$$\xi_m \in \mathbb{R}^{|A|}, \qquad m = 1, \ldots, N_{L+1}$$

we have

$$\lim_{N_1,\ldots,N_L \to \infty} \mathbb{E}\left[\exp\left\{-i \sum_{m=1}^{N_{L+1}} z_m^{(L+1)}(x_A) \cdot \xi_m\right\}\right] = \exp\left\{-\frac{1}{2} \sum_{m=1}^{N_{L+1}} \xi_m^T K_A^{(L)} \xi_m\right\} \tag{3.3.3}$$

with

$$K_A^{(L)} = \left(K^{(L)}(x_\alpha, x_\beta)\right)_{\alpha,\beta \in A}$$

satisfying (3.3.1). To establish (3.3.3) recall that for any $\ell = 1, \ldots, L$

$$z_m^{(\ell+1)}(x_A) \mid z^{(\ell)}(x_A) \text{ are iid centered Gaussian}$$

with covariance $\widehat{K}^{(\ell)}(x_\alpha, x_\beta)$. Thus,

$$\begin{aligned}\mathbb{E}\left[\exp\left\{-i \sum_{m=1}^{N_{L+1}} z_m^{(L+1)}(x_A) \cdot \xi_m\right\}\right] &= \mathbb{E}\left[\mathbb{E}\left[\exp\left\{-i \sum_{m=1}^{N_{L+1}} z_m^{(L+1)}(x_A) \cdot \xi_m\right\} \;\middle|\; z^{(L)}(x_A)\right]\right] \\ &= \mathbb{E}\left[\exp\left\{-\frac{1}{2} \sum_{m=1}^{N_{L+1}} \xi_m^T \widehat{K}_A^{(L)} \xi_m\right\}\right]. \end{aligned} \tag{3.3.4}$$

Theorem 3.3.3 guarantees that the following convergence in distribution:

$$\lim_{n\to\infty} \widehat{K}^{(\ell)}(x_\alpha, x_\beta) = K^{(\ell)}(x_\alpha, x_\beta) := \lim_{n\to\infty} \mathbb{E}\left[\widehat{K}^{(\ell)}(x_\alpha, x_\beta)\right].$$

When combined with (3.3.4) this immediately yields that

$$\lim_{N_1,\ldots,N_L} \mathbb{E}\left[\exp\left\{-i\sum_{m=1}^{N_{L+1}} z_m^{(L+1)}(x_A)\cdot\xi_m\right\}\right] = \exp\left\{-\frac{1}{2}\sum_{m=1}^{N_{L+1}} \xi_m^T K_A^{(L)} \xi_M\right\},$$

where

$$K_A^{(L)} = \left(K^{(L)}(x_\alpha, x_\beta)\right)_{\alpha,\beta\in A}.$$

Thus, we see that $(z_m^{(L+1)}(x_A),\, m = 1,\ldots,N_{L+1})$ indeed converges to independent mean 0 Gaussians with covariance $K_A^{(L)}$. Moreover, for any $\ell = 1,\ldots,L$ we have

$$\begin{aligned} K^{(\ell)}(x_\alpha, x_\beta) &= \lim_{N_1,\ldots,N_{\ell+1}\to\infty} \operatorname{Cov}\left(z_1^{(\ell+1)}(x_\alpha), z_1^{(\ell+1)}(x_\beta)\right) \\ &= \lim_{N_1,\ldots,N_\ell\to\infty} \mathbb{E}\left[\frac{C_W}{N_\ell}\sum_{j=1}^{N_\ell} \sigma\left(z_j^{(\ell)}(x_\alpha)\right)\sigma\left(z_j^{(\ell)}(x_\beta)\right)\right] \\ &= C_W \mathbb{E}_{K^{(\ell-1)}}\left[\sigma\left(z_1(x_\alpha)\right)\sigma\left(z_1(x_\beta)\right)\right], \end{aligned}$$

which confirms (3.3.1). □

3.3.3 Large Depth Asymptotics of the Gaussian Process Limit

In this section, we investigate the effect of depth by first taking $N \to \infty$ and then $L \to \infty$. While we've already seen that the $L, N \to \infty$ limits do not commute in general, we still hope to gain some information about the effect of depth on the properties of randomly initialized networks. We investigate four cases:

- **Linear.** We have according to Theorem 3.3.1

  $$K^{(\ell+1)}(x, x') = C_W \mathbb{E}_{K^{(\ell)}}\left[z_1(x) z_1(x')\right] = C_W K^{(\ell)}(x, x').$$

  Hence, setting $C_W = 1$ shows that the two covariance function for the infinite width Gaussian process is independent of depth.

- **ReLU.** We have according to Theorem 3.3.1

  $$K^{(\ell+1)}(x, x) = C_W \mathbb{E}_{K^{(\ell)}}\left[\sigma(z_1(x))^2\right] = \frac{C_W}{2} K^{(\ell)}(x, x).$$

  Thus, by taking $C_W = 2$, we can keep the variance of the infinite width Gaussian process constant. However, a simple computation (see ?) shows that if we define the correlation operator

  $$C^{(\ell)}(x, x') := \frac{K^{(\ell)}(x, x')}{\sqrt{K^{(\ell)}(x, x) K^{(\ell)}(x', x')}}$$

Then, when $\ell \gg 1$,

$$C^{(\ell)}(x, x') = 1 + O(\ell^{-2}).$$

In other words, the infinite depth limit of the infinite width Gaussian processes for ReLU networks is degenerate in the sense that given any two inputs x, x' of the same norm, the Gaussian field assigns to both of them the same random (Gaussian) constant.

- **Hyperbolic tangent.** Setting $C_W = 1$, we have according to Theorem 3.3.1 that

$$K^{(\ell+1)}(x, x) = \mathbb{E}_{Z\sim\mathcal{N}(0,1)}\left[\sigma\left(Z\sqrt{K^{(\ell)}(x, x)}\right)^2\right], \qquad \sigma(t) := \tanh(t)$$

Since

$$t \neq 0 \quad \Longrightarrow \quad |\sigma(t)| < |t|\,,$$

we find that

$$\forall \epsilon > 0\ \exists \delta > 0 \text{ s.t. } K^{(\ell)}(x, x) > \epsilon \quad \Rightarrow \quad K^{(\ell+1)}(x, x) \leq (1 - \delta)K^{(\ell)}(x, x).$$

So $K^{(\ell)}(x, x)$ rapidly converges at large ℓ to a small neighborhood of 0. Taylor expanding $\tanh(t)$ at $t = 0$ then yields at large ℓ that

$$K^{(\ell+1)}(x, x) \approx K^{(\ell)}(x, x) + 2K^{(\ell)}(x, x)^2 + O\left(K^{(\ell)}(x, x)^3\right).$$

Hence, we see that $K^{(\ell)}(x, x)$ goes to zero like $(2\ell)^{-1}$ at large ℓ (this is made precise in Appendix B of [Han22]). So, just like ReLU (though for different reasons), the Gaussian processes coming from infinite width networks with tanh activations are degenerate at large ℓ.

- **Shaped hyperbolic tangent.** The article [**?**] studies Bayesian inference with the shaped non-linearity

$$\sigma(t) := t + \frac{\psi t}{3L}.$$

Here $\psi \in \mathbb{R}$ is a parameter that controls the strength of the non-linearity. Although each layer is close to linear at large L due to the $1/L$ pre-factor in front of the cubic term, the overall effect is order 1 at the output. The articles [LNR22, MBD$^+$21] show that the $1/L$ scaling is in fact necessary to have a non-degenerate distribution over network outputs at large depth. Indeed, by Theorem 3.3.1 we have

$$K^{(\ell+1)}(x_\mu, x_\nu) = K^{(\ell)}(x_\mu, x_\nu)\left(1 + \frac{\psi}{L}\left(K^{(\ell)}(x_\mu, x_\mu) + K^{(L)}(x_\nu, x_\nu)\right)\right) + O(L^{-2}). \tag{3.3.5}$$

Let us introduce a continuous time index:

$$\ell \quad \mapsto \quad \tau := \ell/L.$$

Taking $x_\mu = x_\nu$ and $\ell \to \infty$ gives

$$\frac{d}{d\tau}K^{(\tau)}(x_\mu, x_\mu) = \frac{2\psi}{L}\left(K^{(\tau)}(x_\mu, x_\mu)\right)^2. \tag{3.3.6}$$

Solving this equation yields

$$K^{(\tau)}(x_\mu, x_\mu) = \left(1 - 2\psi\tau K^{(0)}(x_\mu, x_\mu)\right)^{-1} K^{(0)}(x_\mu, x_\mu)$$
$$= \left(1 - 2\psi\tau \frac{||x_\mu||^2}{N_0}\right)^{-1} \frac{||x_\mu||^2}{N_0}.$$

Plugging this back into (3.3.7) and solving a simple ODE shows that when $L \to \infty$

$$K^{(\tau)}(x_\mu, x_\nu) = \left(1 - 2\psi\tau \frac{||x_\mu||^2}{N_0}\right)^{-1/2} \left(1 - 2\psi\tau \frac{||x_\nu||^2}{N_0}\right)^{-1/2} \frac{1}{N_0} \langle x_\mu, x_\nu \rangle . \tag{3.3.7}$$

Hence, the variance and covariance of the Gaussian process are non-degenerate at large L.

3.4 Finite Width Corrections to the Gaussian Process Limit

In this section, we will explore explain, in the simplest possible instance, how to study perturbative corrections in powers of $1/N$ to the Gaussian process limit $N \to \infty$ of random neural networks. This has been taken up in a range of prior works [Han18, HR18, HN20a, HN20b, **?**, RYH22, Yai20].

To start let's consider the recursion relation (3.3.1), which describes the infinite width Gaussian Process covariance $K^{(\ell+1)}$ in terms of C_W, σ and $K^{(\ell)}$. As shown most fully in [Han22, RYH22] this recursion is the beginning of a perturbatively solvable hierarchy of recursions that can be used to describe virtually every observable associated with a random neural network. This includes the higher cumulants of the functions computed by network neurons, and we'll see an example below of how cumulants of order k in layer $\ell + 1$ are determined only via cumulants of order $j \leq k$ at layer ℓ. We will focus on explaining this in the context of perhaps the simplest deviation to the Gaussian process behavior at infinite width, given by the fourth cumulant at a single input

$$\kappa_4^{(\ell)}(x) = \frac{1}{3}\kappa\left(z_1^{(\ell)}(x), z_1^{(\ell)}(x), z_1^{(\ell)}(x), z_1^{(\ell)}(x)\right) = \frac{1}{3}\left(\mathbb{E}\left[\left(z_1^{(\ell)}(x)\right)^4\right] - 3\mathbb{E}\left[\left(z_1^{(\ell)}(x)\right)^2\right]^2\right).$$

A simple computation shows that $\kappa_4^{(\ell)}(x)$ captures both non-Gaussian fluctuations

$$\mathrm{Var}\left[\left(z_i^{(\ell)}(x)\right)^2\right] = 3\kappa_4^{(\ell)}(x) + 2\mathbb{E}\left[\left(z_i^{(\ell)}(x)\right)^2\right]^2$$

and inter-neuron correlations

$$\kappa_4^{(\ell)}(x) = \mathrm{Cov}\left(\left(z_i^{(\ell)}(x)\right)^2, \left(z_j^{(\ell)}(x)\right)^2\right), \qquad i \neq j.$$

Since as $N_1, \cdots, N_L \to \infty$, neurons are independent and Gaussian, we have that

$$\lim_{N_1,\ldots,N_{\ell-1}\to\infty} \kappa_4^{(\ell)}(x) = 0.$$

The following result shows that, $\kappa_4^{(\ell)}(x)$ has order depth/width. This is an example of how the large depth and large width limits do not commute, with depth accentuating finite-width effects.

Theorem 3.4.1. *Fix L, N_0, N_{L+1}, σ. Suppose that the weights and biases are chosen as in* (**??**) *and that*

$$N_1, \ldots, N_L \simeq N \gg 1.$$

The fourth cumulant is of order $O(N^{-1})$ and satisfies the following recursion:

$$\kappa_4^{(\ell+1)}(x) = \frac{C_W^2}{N_\ell}\mathrm{Var}_{K^{(\ell)}}\left[\sigma^2\right] + \left(C_W\mathbb{E}_{K^{(\ell)}}\left[\frac{\partial^2}{\partial t^2}\sigma^2(t)\right]\right)^2 \kappa_4^{(\ell)}(x) + O(N^{-2}). \tag{3.4.1}$$

Thus, choosing C_W to be "tuned to criticality", i.e. such that

$$C_W\mathbb{E}_{K^{(\ell)}}\left[\frac{\partial^2}{\partial t^2}\sigma^2(t)\right] = 1,$$

and taking $N_\ell = N$, we have

$$\frac{\kappa_4^{(L+1)}(x)}{\left(K^{(L+1)}(x,x)\right)^2} = C_\sigma\frac{L}{N} + O_{L,\sigma}(N^{-2}).$$

Moreover, for any fix $m \geq 1$ and any "reasonable" function $f : \mathbb{R}^m \to \mathbb{R}$ we may write

$$\begin{aligned}
&\mathbb{E}\left[f\left(z_i^{(\ell)}(x),\, 1 \leq i \leq m\right)\right] \\
&\quad = \mathbb{E}_{G^{(\ell)}}\left[f\left(z_i^{(\ell)}(x),\, 1 \leq i \leq m\right)\right] \\
&\quad + \frac{\kappa_4^{(\ell+1)}(x)}{8}\mathbb{E}_{K^{(\ell)}}\left[\left(\sum_{j=1}^m \partial_{z_j}^4 + \sum_{\substack{j_1,j_2=1 \\ j_1\neq j_2}}^m \partial_{z_{j_1}}^2\partial_{z_{j_2}}^2\right) f\left(z_i^{(\ell)}(x),\, 1 \leq i \leq m\right)\right] \\
&\quad + O(N^{-2}).
\end{aligned} \tag{3.4.2}$$

Here, $G^{(\ell)}$ is the dressed two point function

$$G^{(\ell)}(x_\alpha, x_\beta) := \mathbb{E}\left[z_1^{(\ell)}(x_\alpha)z_1^{(\ell)}(x_\beta)\right].$$

This Theorem is originally derived in a physics way in the breakthrough paper of Yaida [Yai20]. It was then rederived, again at a physics level of rigor in Chapter 4 of [RYH22]. Finally, it was derived in a somewhat different, and more mathematical, way in [Han22]. We give the sketch of proof in §3.4.1.

3.4.1 Proof Sketch of Theorem 3.4.1

For this proof, since we will suppress the network input x from out notation. So for instance $\widehat{K}^{(\ell)}(x,x)$ will simply be denoted by $\widehat{K}^{(\ell)}$. Since $\widehat{K}^{(\ell)}$ is a collective observable, it makes sense to consider

$$G^{(\ell)} := \mathbb{E}\left[\widehat{K}^{(\ell)}\right], \qquad \Delta^{(\ell-1)} := \widehat{K}^{(\ell)} - \mathbb{E}\left[\widehat{K}^{(\ell)}\right].$$

The scalar $G^{(\ell)}$ is sometimes referred to as a dressed two point function. Note that

$$\kappa_4^{(\ell)} = \mathbb{E}\left[\left(\Delta^{(\ell-1)}\right)^2\right]$$

Just as before, we have

$$\mathbb{E}\left[f(z_i^{(\ell)},\, i=1,\ldots,m)\right] = \int_{\mathbb{R}^{n_m}} \widehat{f}\,(\xi)\,\mathbb{E}\left[e^{-\frac{1}{2}||\xi||^2\widehat{K}^{(\ell)}}\right] d\xi$$
$$= \int_{\mathbb{R}^{n_m}} \widehat{f}\,(\xi)\, e^{-\frac{1}{2}||\xi||^2 G^{(\ell)}}\mathbb{E}\left[e^{-\frac{1}{2}||\xi||^2\Delta^{(\ell-1)}}\right] d\xi.$$

Applying Theorem 3.3.3 we may actually Taylor expand to find a power series expansion in $1/N$:

$$\mathbb{E}\left[e^{-\frac{1}{2}||\xi||^2\Delta^{(\ell-1)}}\right] = \sum_{q\geq 0}\frac{(-1)^q}{2^q q!}\,||\xi||^{2q}\,\mathbb{E}\left[\left(\Delta^{(\ell-1)}\right)^q\right] = 1 + \frac{1}{8}\,||\xi||^4\,\mathbb{E}\left[\left(\Delta^{(\ell-1)}\right)^2\right] + O(N^{-2}).$$

Putting this all together yields

$$\mathbb{E}\left[f(z_i^{(\ell)},\, i=1,\ldots,m)\right] = \int_{\mathbb{R}^{n_m}}\left(1 + \frac{1}{8}\,||\xi||^4\,\mathbb{E}\left[\left(\Delta^{(\ell-1)}\right)^2\right]\right)\widehat{f}\,(\xi)\, e^{-\frac{1}{2}||\xi||^2 G^{(\ell)}} d\xi + O(N^{-2}).$$

In particular, we obtain

$$\mathbb{E}\left[f(z_i^{(\ell)},\, i=1,\ldots,m)\right] = \mathbb{E}_{G^{(\ell)}}\left[f\right] + \frac{1}{8}\mathbb{E}\left[\left(\Delta^{(\ell-1)}\right)^2\right]\mathbb{E}_{G^{(\ell)}}\left[\left(\sum_{j=1}^{m}\partial^2_{z_j^{(\ell)}}\right)^2 f\right] + O(N^{-2}). \tag{3.4.3}$$

A direct computation now shows that

$$\mathbb{E}_{G^{(\ell)}}\left[f\right] = \mathbb{E}_{K^{(\ell)}}\left[f\right] + O(N^{-1}).$$

This proves (3.4.2). Next, recall that

$$\kappa_4^{(\ell+1)}(x) = \mathbb{E}\left[\left(\Delta^{(\ell)}\right)^2\right].$$

Moreover,

$$\mathbb{E}\left[\left(\Delta^{(\ell)}\right)^2\right] = \frac{1}{n_\ell}\mathbb{E}\left[\left(X_{1;\alpha}^{(\ell)}\right)^2\right] + \left(1 - \frac{1}{n_\ell}\right)\mathbb{E}\left[X_{1;\alpha}^{(\ell)}X_{2;\alpha}^{(\ell)}\right],$$

where

$$X_{j;\alpha}^{(\ell)} := C_W\left(\sigma(z_j^{(\ell)})^2 - \mathbb{E}\left[\sigma(z_j^{(\ell)})^2\right]\right).$$

Applying (3.4.3) and some algebra completes the proof of (3.4.1). □

Bibliography

Effect of depth on first step of training:

- ReLU EVGP
- Change in NTK
- Quantitative CLTs
- Diagramatics of Banta

4 Nodal Set Problems from Random ReLU Networks

In this chapter, we briefly survey some problems concerning nodal sets generated by randomly initialized ReLU networks. This subject seems to have been first considered in [HR19, HR], and we mainly follow the exposition in these articles. Specifically, let us consider fully connected depth L network

$$x \mapsto z^{(L+1)}(x;\theta) \in \mathbb{R}^{N_{L+1}}$$

with input dimension N_0, hidden layer widths $N_1, \dots, N_L$, output dimension N_{L+1} and ReLU activations, defined by

$$z_i^{(\ell+1)}(x) = \begin{cases} b_i^{(\ell+1)} + \frac{1}{\sqrt{N_\ell}} \sum_{j=1}^{N_\ell} W_{ij}^{(\ell)} \sigma\left(z_j^{(\ell)}(x)\right), & \ell \geq 0 \\ b_i^{(1)} + \frac{1}{\sqrt{N_0}} \sum_{j=1}^{N_0} W_{ij}^{(1)} x_j, & \ell = 0 \end{cases}, \tag{4.0.1}$$

where $i = 1, \dots, N_{\ell+1}$ and

$$\sigma(t) = \mathrm{ReLU}(t) = \max\{0, t\}.$$

This definition of a ReLU network differs slightly from our standing definition (2.1.6) in that we've added to the parameter vector θ the network biases $b_i^{(\ell)}$, which in much of the other Chapters we have set to zero. Throughout this chapter we will make the following

$$\textbf{assumption}: \quad \text{the output dimension } N_{L+1} = 1.$$

4.1 Nodal Sets and Nodal Partitions

For any setting of weights and biases θ, the map $x \in \mathbb{R}^{N_0} \mapsto z^{(L+1)}(x;\theta) \in \mathbb{R}$ is continuous and piecewise linear. This means that each θ determines a partition of the input space $\mathbb{R}^{N_0}$ into disjoint polyhedra, such that on each such polyhedron the map $x \mapsto z^{(L+1)}(x;\theta)$ is affine. We will call this collection of polyhedra the *nodal partition* associated to a ReLU network.

A well-known observation is that the partition generated by ReLU networks with one hidden layer of size N_1 and input dimension N_0 are precisely those arising as the cells in an arrangements of N_1 hyperplanes in $\mathbb{R}^{N_0}$[1]. Indeed, the function computed by such a network is

$$z^{(2)}(x;\theta) = b^{(2)} + \sum_{i=1}^{N_1} W_i^{(2)} \sigma\left(z_i^{(1)}(x)\right), \quad z_i^{(1)}(x) = b_i^{(1)} + W_i^{(1)} \cdot x. \tag{4.1.1}$$

The i^{th} neuron $z_i^{(1)}(x)$ determines the (co-oriented) hyperplane

$$H_i^{(1)} := \left\{ x \in \mathbb{R}^{N_0} \mid z_i^{(1)}(x) = W_i^{(1)} \cdot x + b_i^{(1)} = 0 \right\}.$$

[1] Recall that the cells in an arrangement of co-dimension 1 hyperplanes are the connected components of the complement of the union of these hyperplanes

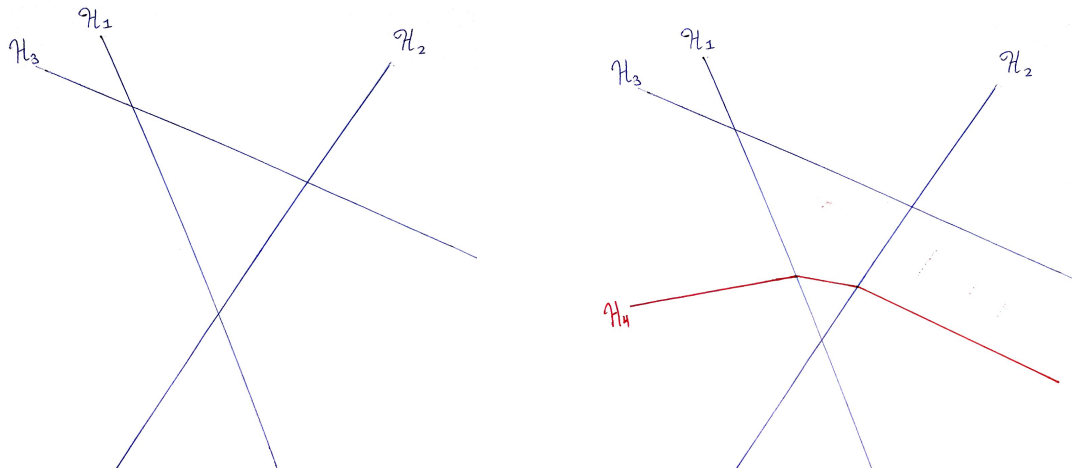

Figure 4.1: In the left figure, the partition of inputs space induced by a depth 1 ReLU network with three neurons z_1, z_2, z_3 in its hidden layer. Each neuron z_j corresponds to a hyperplane H_j at which its ReLU turns from on to off. In the right figure, one connected component of the "bent hyperplane" coming from a neuron in a subsequent layer it shown.

For inputs x lying on one side of $H_i^{(1)}$, the neuron $z_i^{(1)}$ is "on" in the sense that its post-activation $\sigma\left(z_i^{(1)}(x)\right)$ is the affine function $W_i^{(1)} \cdot x + b_i^{(1)}$, whereas for inputs on the other side it is "off" in the sense that its post-activation is identically 0. On each cell of the hyperplane arrangement $\left\{H_1^{(1)}, \dots, H_n^{(1)}\right\}$, the collection of neurons that are on is fixed and hence $x \mapsto z^{(2)}(x;\theta)$ restricted to such a cell is an affine function. This is illustrated in Figure 4.1.

For deeper ReLU networks we can still define the nodal set

$$H_i^{(\ell)} := \left\{x \in \mathbb{R}^{N_0} \mid z_i^{(\ell)}(x) = 0\right\}$$

of the i^{th} neuron in the ℓ^{th} layer ℓ pre. When $\ell > 1$, however, the sets $H_i^{(\ell)}$ are in general no longer hyperplanes. Instead, they can intuitively be thought of as collections of "bent hyperplanes," composed of a finite number of connected components, each of which is piecewise linear (i.e. locally looks like a hyperplane). For instance, the pre-activation $z_i^{(2)}(x)$ of a neuron in the second hidden layer has exactly the same for as the output of a one hidden layer ReLU network. In particular, it is given by a different affine function on each cell generated by the arrangement of hyperplanes $H_i^{(1)}$, $i = 1, \dots, N_1$, from neurons in the first layer (displayed on the left in Figure 4.1). The equation for $z_i^{(2)}(x) = 0$ is therefore a different linear equation on each such cell and depends on which neurons in the first layer are on for the input x (see the right panel in Figure 4.1).

We will call the union of $H_i^{(\ell)}$ over all $\ell = 1, \dots, L$ and all $i = 1, \dots, N_\ell$ the *nodal set* of the ReLU network. Similarly, we define the *nodal partition* of a ReLU network to be the connected components of the complement of the union of the $H_i^{(\ell)}$'s.

4.2 Geometry and Topology ReLU Nodal Sets and Partitions

A natural question from the point of view of the study of nodal sets of random functions is to understand the size and shape of the nodal sets and nodal partitions of a ReLU network with random weights and biases. Concretely, let $x \mapsto z^{(L+1)}(x;\theta)$ be a ReLU network with input dimension N_0, hidden layer widths $N_1, \dots, N_L$, and output dimension $N_{L+1} = 1$ as in (4.0.1). Suppose moreover that the weights and biases are chosen at random as follows:

$$W_{ij}^{(\ell)} \sim \mathcal{N}(0,2),\, b_i^{(\ell)} \sim \mathcal{N}(0,1) \quad \text{independent}, \qquad 1 \leq \ell \leq L+1,\, 1 \leq i \leq N_\ell.$$

The factor of 2 in the variance scaling for the weights is necessary to ensure that the average squared input-output Jacobian $\mathbb{E}\left[\left|\left|\nabla_x z^{(L+1)}(x)\right|\right|^2\right] = 1$ for every non-zero network input (see §**??**). The following are examples of natural geometric/topological questions about the nodal set and nodal partition of such a random ReLU network:

- **Q1.** (Nodal set volume) For any compact set $A \subseteq \mathbb{R}^{N_0}$, what is the distribution of the random variable
$$Z_A := \mathrm{vol}\left(A \cap \bigcup_{\substack{\ell=1,\dots,L \\ i=1,\dots,N_\ell}} \mathcal{H}_i^{(\ell)}\right),$$
where vol denotes the (N_0-1)-dimensional Hausdorff measure?

- **Q2.** (Nodal set topology) For any set $A \subseteq \mathbb{R}^{N_0}$, what is the distribution of the random variable
$$N_A := \#\left\{\text{connected components of } \mathbb{R}^{N_0} \setminus \bigcup_{\substack{\ell=1,\dots,L \\ i=1,\dots,N_\ell}} \mathcal{H}_i^{(\ell)} \text{ that intersect } A\right\}?$$

- **Q3.** (Partial nodal set topology) Fix a layer index $1 \leq \ell \leq L+1$ and a neural index $1 \leq i \leq N_\ell$, and a subset $A \subseteq \mathbb{R}^{N_0}$. What is the distribution of the random variable
$$C_A := \#\left\{\text{connected components of } H_i^{(\ell)} \text{ that intersect } A\right\}?$$

There is essentially nothing known about Q3. For Q1 and Q2, the articles [HR19, HR] show that, somewhat surprisingly, the means of these random variance depend very weakly on the network depth. For example, we have the following:

Theorem 4.2.1 ([HR19])**.** *Consider a random ReLU network with input dimension N_0, hidden layer widths $N_1, \dots, N_L$ as in* (4.0.1)*. There exists a constant $C > 0$ depending only on the sum of the reciprocals $N_1^{-1} + \cdots + N_L^{-1}$ of the hidden layer widths such that for every $A \subseteq \mathbb{R}^{N_0}$*

$$\mathbb{E}\left[Z_A\right] \leq C\left(N_1 + \cdots + N_L\right)\mathrm{vol}(A), \tag{4.2.1}$$

where $\mathrm{vol}(A)$ *is the Euclidean volume of A and $N_1 + \cdots + N_L$ is the number of neurons in the network.*

The remarkable aspect of the estimate (4.2.1) is that the upper bound depends on the network architecture only through the number of total neurons and not on whether they are arranged into a single hidden layer or into a deeper network (under the assumption that the the sum of reciprocals of the hidden layer widths is bounded). Note that in the case of one hidden layer network, the random variable Z_A is simply a sum of iid positive random variables:

$$Z_A = \sum_{i=1}^{N_1} \mathrm{vol}\left(A \cap H_i^{(1)}\right).$$

In this case, it is easy to see that the variance is small relative to the mean. It is conjectured that the variance of Z_A grows with the network depth. It is an interesting open problem to prove or disprove this conjecture.

The proof of Theorem is a more or less straight-forward application of the co-area formula [**?**] (see ...). Instead of reproducing it, we provide here a simple intuition for why the estimate (4.2.1) ought to hold when $N_0 = 1$. In this case, we have

$$Z_A = \sum_{\ell=1}^{L} \sum_{i=1}^{N_L} \#\left\{x \in \mathbb{R} \mid z_i^{(\ell)}(x) = 0\right\}.$$

The estimate (4.2.1) then follows once we establish that

$$\mathbb{E}\left[\#\left\{x \in \mathbb{R} \mid z_i^{(\ell)}(x) = 0\right\}\right] = O(1), \tag{4.2.2}$$

where the implicit constant depends only on the sum of the reciprocals of the hidden layer widths. Writing

$$\widehat{z}_i^{(\ell)}(x) := \sum_{j=1}^{N_{\ell-1}} W_{ij}^{(\ell)} \sigma\left(z_j^{(\ell)}(x)\right),$$

we have that

$$z_i^{(\ell)}(x) = 0 \qquad \Longleftrightarrow \qquad \widehat{z}_i^{(\ell)}(x) = -b_i^{(\ell)}.$$

Note that $x \mapsto \widehat{z}_i^{(\ell)}(x)$ is a random continuous piecewise linear function. Moreover, by [**?**],?, we have

$$\mathbb{E}\left[\left|\left|\nabla_x \widehat{z}_i^{(\ell)}(x)\right|\right|^2\right] = 1.$$

Thus, while $x \mapsto \widehat{z}_i^{(\ell)}(x)$ is not affine, it also can't many large oscillations. In particular, we expect that it can cross the random level $-b_i^{(\ell)}$ only a finite number of times, which would exactly imply (4.2.2).

To complete our discussion of nodal sets of random ReLU networks, let us note that when the input dimension N_0 equals 1, the nodal set of a ReLU network is typically a discrete number of points. Hence, the number of connected components of the nodal partition, which are the intervals between these points, is exactly one more than the co-dimension volume (i.e. the number) of points in the nodal set. In higher input dimension the relationship between the number of connected components in the nodal partition and the volume of the nodal set is a bit more subtle, and virtually the only result about the number of connected components is the following:

Theorem 4.2.2 ([HR])**.** *Consider the same assumptions and notation as in Theorem 4.2.1. There exists constants $\delta, C > 0$ such that in any cube $\mathcal{C}$ of side-length at least δ the average number of connected components for the nodal partition of a random ReLU network is at most the volume of $\mathcal{C}$ times*

$$\min\left\{2^{(N_1+\cdots+N_L)}, \frac{1}{N_0!}\left(C\left(N_1+\cdots+N_L\right)^{N_0}\right)\right\}.$$

By Zaslavsky's Theorem on the number of cells in a hyperplane arrangement, we have that when $L = 1$, with probability 1, total number of connected components in the nodal partition of a one layer ReLU network is

$$\sum_{k=0}^{N_0}\binom{N_1}{k} \simeq \min\left\{2^{N_1}, \frac{N_1^{N_0}}{N_0!}\right\}.$$

Thus, just as in Theorem 4.2.1, Theorem 4.2.2 suggests that, at least on average, deeper networks do not have a significantly large number of connected components per neuron in their nodal partitions.

Bibliography

[AAM22] Emmanuel Abbe, Enric Boix Adsera, and Theodor Misiakiewicz. The merged-staircase property: a necessary and nearly sufficient condition for sgd learning of sparse functions on two-layer neural networks. In *Conference on Learning Theory*, pages 4782–4887. PMLR, 2022.

[AAM23] Emmanuel Abbe, Enric Boix Adsera, and Theodor Misiakiewicz. Sgd learning on neural networks: leap complexity and saddle-to-saddle dynamics. In *The Thirty Sixth Annual Conference on Learning Theory*, pages 2552–2623. PMLR, 2023.

[ACGH19] Sanjeev Arora, Nadav Cohen, Noah Golowich, and Wei Hu. A convergence analysis of gradient descent for deep linear neural networks. *ICLR*, 2019.

[ACHL19] Sanjeev Arora, Nadav Cohen, Wei Hu, and Yuping Luo. Implicit regularization in deep matrix factorization. In *Advances in Neural Information Processing Systems*, pages 7413–7424, 2019.

[ADH+19a] Sanjeev Arora, Simon Du, Wei Hu, Zhiyuan Li, and Ruosong Wang. Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks. In *International Conference on Machine Learning*, pages 322–332. PMLR, 2019.

[ADH+19b] Sanjeev Arora, Simon S Du, Wei Hu, Zhiyuan Li, Russ R Salakhutdinov, and Ruosong Wang. On exact computation with an infinitely wide neural net. In *Advances in Neural Information Processing Systems*, pages 8139–8148, 2019.

[AGJ21] Gerard Ben Arous, Reza Gheissari, and Aukosh Jagannath. Online stochastic gradient descent on non-convex losses from high-dimensional inference. *Journal of Machine Learning Research*, 22(106):1–51, 2021.

[AGZ10] Greg W Anderson, Alice Guionnet, and Ofer Zeitouni. *An introduction to random matrices*. Number 118. Cambridge university press, 2010.

[AKLS23] Luca Arnaboldi, Florent Krzakala, Bruno Loureiro, and Ludovic Stephan. Escaping mediocrity: how two-layer networks learn hard single-index models with sgd. *arXiv preprint arXiv:2305.18502*, 2023.

[B+12] Philippe Bougerol et al. *Products of random matrices with applications to Schrödinger operators*, volume 8. Springer Science & Business Media, 2012.

[BAGJ22] Gerard Ben Arous, Reza Gheissari, and Aukosh Jagannath. High-dimensional limit theorems for sgd: Effective dynamics and critical scaling. *Advances in Neural Information Processing Systems*, 35:25349–25362, 2022.

[Bar92] Andrew R Barron. Neural net approximation. In *Proc. 7th Yale Workshop on Adaptive and Learning Systems*, volume 1, pages 69–72, 1992.

[BBPV23] Alberto Bietti, Joan Bruna, and Loucas Pillaud-Vivien. On learning gaussian multi-index models with gradient flow. *arXiv preprint arXiv:2310.19793*, 2023.

[BBSS22] Alberto Bietti, Joan Bruna, Clayton Sanford, and Min Jae Song. Learning single-index models with shallow neural networks. *Advances in Neural Information Processing Systems*, 35:9768–9783, 2022.

[BH89] Pierre Baldi and Kurt Hornik. Neural networks and principal component analysis: Learning from examples without local minima. *Neural networks*, 2(1):53–58, 1989.

[BLLT20] Peter L Bartlett, Philip M Long, Gábor Lugosi, and Alexander Tsigler. Benign overfitting in linear regression. *Proceedings of the National Academy of Sciences*, 2020.

[BMR21] Peter L Bartlett, Andrea Montanari, and Alexander Rakhlin. Deep learning: a statistical viewpoint. *arXiv preprint arXiv:2103.09177*, 2021.

[BMZ23] Raphaël Berthier, Andrea Montanari, and Kangjie Zhou. Learning time-scales in two-layers neural networks. *arXiv preprint arXiv:2303.00055*, 2023.

[BNL+23] Blake Bordelon, Lorenzo Noci, Mufan Bill Li, Boris Hanin, and Cengiz Pehlevan. Depthwise hyperparameter transfer in residual networks: Dynamics and scaling limit. *arXiv preprint arXiv:2309.16620*, 2023.

[BP22] Blake Bordelon and Cengiz Pehlevan. Self-consistent dynamical field theory of kernel evolution in wide neural networks. *Advances in Neural Information Processing Systems*, 35:32240–32256, 2022.

[CB18a] Lenaic Chizat and Francis Bach. A note on lazy training in supervised differentiable programming. *arXiv preprint arXiv:1812.07956*, 2018.

[CB18b] Lenaic Chizat and Francis Bach. On the global convergence of gradient descent for over-parameterized models using optimal transport. In *Advances in neural information processing systems*, pages 3036–3046, 2018.

[CWPPS23] Elizabeth Collins-Woodfin, Courtney Paquette, Elliot Paquette, and Inbar Seroussi. Hitting the high-dimensional notes: An ode for sgd learning dynamics on glms and multi-index models. *arXiv preprint arXiv:2308.08977*, 2023.

[DHP21] Ronald DeVore, Boris Hanin, and Guergana Petrova. Neural network approximation. *Acta Numerica*, 30:327–444, 2021.

[DLT+18] Simon S Du, Jason D Lee, Yuandong Tian, Barnabas Poczos, and Aarti Singh. Gradient descent learns one-hidden-layer cnn: Don't be afraid of spurious local minima. *ICML*, 2018.

[DZPS19] Simon S. Du, Xiyu Zhai, Barnabas Poczos, and Aarti Singh. Gradient descent provably optimizes over-parameterized neural networks. In *International Conference on Learning Representations*, 2019.

[FK60] Harry Furstenberg and Harry Kesten. Products of random matrices. *The Annals of Mathematical Statistics*, 31(2):457–469, 1960.

[GARA18] Adrià Garriga-Alonso, Carl Edward Rasmussen, and Laurence Aitchison. Deep convolutional networks as shallow gaussian processes. *arXiv preprint arXiv:1808.05587*, 2018.

[GLSS18] Suriya Gunasekar, Jason D Lee, Daniel Soudry, and Nati Srebro. Implicit bias of gradient descent on linear convolutional networks. In *Advances in Neural Information Processing Systems*, pages 9461–9471, 2018.

[GWB+18] Suriya Gunasekar, Blake Woodworth, Srinadh Bhojanapalli, Behnam Neyshabur, and Nathan Srebro. Implicit regularization in matrix factorization. In *2018 Information Theory and Applications Workshop (ITA)*, pages 1–10. IEEE, 2018.

[Han18] Boris Hanin. Which neural net architectures give rise to exploding and vanishing gradients? In *Advances in Neural Information Processing Systems*, 2018.

[Han22] Boris Hanin. Random fully connected neural networks as perturbatively solvable hierarchies. *arXiv preprint arXiv:2204.01058*, 2022.

[Han23] Boris Hanin. Random neural networks in the infinite width limit as gaussian processes. *Annals of Applied Probability (to appear). arXiv:2107.01562*, 2023.

[HBSDN20] Jiri Hron, Yasaman Bahri, Jascha Sohl-Dickstein, and Roman Novak. Infinite attention: Nngp and ntk for deep attention networks. In *International Conference on Machine Learning*, pages 4376–4386. PMLR, 2020.

[HN20a] Boris Hanin and Mihai Nica. Finite depth and width corrections to the neural tangent kernel. *ICLR 2020*, 2020.

[HN20b] Boris Hanin and Mihai Nica. Products of many large random matrices and gradients in deep neural networks. *Communications in Mathematical Physics*, 376(1):287–322, 2020.

[HR] Boris Hanin and David Rolnick. Deep relu networks have surprisingly few activation patterns. *NeurIPS 2019*.

[HR18] Boris Hanin and David Rolnick. How to start training: The effect of initialization and architecture. In *Advances in Neural Information Processing Systems*, pages 571–581, 2018.

[HR19] Boris Hanin and David Rolnick. Complexity of linear regions in deep networks. *ICML*, 2019.

[HSW89] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989.

[JGH18] Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in neural information processing systems*, pages 8571–8580, 2018.

[JGN+17] Chi Jin, Rong Ge, Praneeth Netrapalli, Sham M Kakade, and Michael I Jordan. How to escape saddle points efficiently. *arXiv preprint arXiv:1703.00887*, 2017.

[JT20] Ziwei Ji and Matus Telgarsky. Directional convergence and alignment in deep learning. *Advances in Neural Information Processing Systems*, 33:17176–17186, 2020.

[Kaw16] Kenji Kawaguchi. Deep learning without poor local minima. In *Advances in Neural Information Processing Systems*, pages 586–594, 2016.

[LBD+20] Aitor Lewkowycz, Yasaman Bahri, Ethan Dyer, Jascha Sohl-Dickstein, and Guy Gur-Ari. The large learning rate phase of deep learning: the catapult mechanism. *arXiv preprint arXiv:2003.02218*, 2020.

[LBN+18] Jaehoon Lee, Yasaman Bahri, Roman Novak, Samuel S Schoenholz, Jeffrey Pennington, and Jascha Sohl-Dickstein. Deep neural networks as gaussian processes. *ICML 2018 andarXiv:1711.00165*, 2018.

[LGJ20] Meng Liu, Hongyang Gao, and Shuiwang Ji. Towards deeper graph neural networks. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 338–348, 2020.

[LK17] Haihao Lu and Kenji Kawaguchi. Depth creates no bad local minima. *arXiv preprint arXiv:1702.08580*, 2017.

[LL19] Kaifeng Lyu and Jian Li. Gradient descent maximizes the margin of homogeneous neural networks. In *International Conference on Learning Representations*, 2019.

[LNR22] Mufan Bill Li, Mihai Nica, and Daniel M Roy. The neural covariance sde: Shaped infinite depth-and-width networks at initialization. *NeurIPS 2022*, 2022.

[LZB22a] Chaoyue Liu, Libin Zhu, and Mikhail Belkin. Loss landscapes and optimization in over-parameterized non-linear systems and neural networks. *Applied and Computational Harmonic Analysis*, 59:85–116, 2022.

[LZB22b] Chaoyue Liu, Libin Zhu, and Mikhail Belkin. Transition to linearity of wide neural networks is an emerging property of assembling weak models. *arXiv preprint arXiv:2203.05104*, 2022.

[MBD+21] James Martens, Andy Ballard, Guillaume Desjardins, Grzegorz Swirszcz, Valentin Dalibard, Jascha Sohl-Dickstein, and Samuel S Schoenholz. Rapid training of deep neural networks without skip connections or normalization layers using deep kernel shaping. *arXiv preprint arXiv:2110.01765*, 2021.

[MHPG+22] Alireza Mousavi-Hosseini, Sejun Park, Manuela Girotti, Ioannis Mitliagkas, and Murat A Erdogdu. Neural networks efficiently learn low-dimensional representations with sgd. *arXiv preprint arXiv:2209.14863*, 2022.

[MMN18] 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.

[MP16] Hrushikesh N Mhaskar and Tomaso Poggio. Deep vs. shallow networks: An approximation theory perspective. *Analysis and Applications*, 14(06):829–848, 2016.

[MS17] James A Mingo and Roland Speicher. *Free probability and random matrices*, volume 35. Springer, 2017.

[Nea96] Radford M Neal. Priors for infinite networks. In *Bayesian Learning for Neural Networks*, pages 29–53. Springer, 1996.

[OS20] Samet Oymak and Mahdi Soltanolkotabi. Toward moderate overparameterization: Global convergence guarantees for training shallow neural networks. *IEEE Journal on Selected Areas in Information Theory*, 1(1):84–105, 2020.

[PN21] Huy Tuan Pham and Phan-Minh Nguyen. Global convergence of three-layer neural networks in the mean field regime. *ICLR*, 2021.

[RVE18] Grant Rotskoff and Eric Vanden-Eijnden. Parameters as interacting particles: long time convergence and asymptotic error scaling of neural networks. *Advances in neural information processing systems*, 31, 2018.

[RYH22] Daniel A Roberts, Sho Yaida, and Boris Hanin. *The Principles of Deep Learning Theory: An Effective Theory Approach to Understanding Neural Networks*. Cambridge University Press, 2022.

[SHN+18] Daniel Soudry, Elad Hoffer, Mor Shpigel Nacson, Suriya Gunasekar, and Nathan Srebro. The implicit bias of gradient descent on separable data. *The Journal of Machine Learning Research*, 19(1):2822–2878, 2018.

[SJL18] Mahdi Soltanolkotabi, Adel Javanmard, and Jason D Lee. Theoretical insights into the optimization landscape of over-parameterized shallow neural networks. *IEEE Transactions on Information Theory*, 65(2):742–769, 2018.

[SS20] Justin Sirignano and Konstantinos Spiliopoulos. Mean field analysis of neural networks: A central limit theorem. *Stochastic Processes and their Applications*, 130(3):1820–1852, 2020.

[SS21] Justin Sirignano and Konstantinos Spiliopoulos. Mean field analysis of deep neural networks. *Mathematics of Operations Research*, 2021.

[WGL+20] Blake Woodworth, Suriya Gunasekar, Jason D Lee, Edward Moroshko, Pedro Savarese, Itay Golan, Daniel Soudry, and Nathan Srebro. Kernel and rich regimes in overparametrized models. *arXiv preprint arXiv:2002.09277*, 2020.

[Yai20] Sho Yaida. Non-gaussian processes and neural networks at finite widths. *MSML*, 2020.

[Yan19] Greg Yang. Tensor programs i: Wide feedforward or recurrent neural networks of any architecture are gaussian processes. *arXiv preprint arXiv:1910.12478*, 2019.

[YH21] Greg Yang and Edward J Hu. Tensor programs iv: Feature learning in infinite-width neural networks. In *International Conference on Machine Learning*, pages 11727–11737. PMLR, 2021.

[ZBH+17] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. In *International Conference on Learning Representations, (ICLR)*, 2017.