Neural Networks at Large Width and Depth

Boris Hanin
Princeton ORFE

January 8, 2023

Abstract

These are notes roughly follow a mini-course given by the author in the Rome Center on Mathematics for Modeling and Data Science at Tor Vergata in January 2023.

Contents

1 Introduction **2**
1.1 What is a (Random) Neural Network? . . . 2
1.2 Two Examples of Functions Computed by a Neural Network . . . 2
1.3 Typical Use of Neural Networks . . . 3
1.4 Broad Questions . . . 4
1.5 Goal for Lectures 2, 3 . . . 5

2 Infinite Width Networks at Fixed Depth: GP/NTK Regime **7**
2.1 Random Neural Networks . . . 7
2.2 Statement of GP Limit at Infinite Width and Finite Depth . . . 7
2.3 Structural Properties of Random Neural Networks at Large Width . . . 8
2.4 Derivation of GP Limit: Proof of Theorem 2.1 . . . 8
2.5 Optimization of Wide Networks in the NTK Regime . . . 10
2.5.1 Idea 1 in Theorem 2.4: Tangent Kernel . . . 10
2.5.2 Idea 2 in Theorem 2.4: Linearization at Large Width . . . 12

3 The Importance of Finite Width Effects at Large Depth **13**
3.1 Intuition for Appearance of L/n . . . 15
3.2 Proof Sketch of Theorem 3.1 . . . 16
3.3 Tuning to Criticality: Analysis of GP Limit at Large Depth . . . 17
3.3.1 The Case of Linear Activations. . . . 18
3.3.2 Tuning to Criticality: Setting C_b, C_W for General Activations . . . 18
3.3.3 Criticality for ReLU . . . 18
3.4 Extensions and Open Problems . . . 18

1 Introduction

1.1 What is a (Random) Neural Network?

At a high level a neural network is a parameterized family of function. We will focus on perhaps the simplest, so-called fully connected, networks.

Definition 1.1. *Fix $L, n_0, \dots, n_{L+1} \geq 1$ and $\sigma : \mathbb{R} \to \mathbb{R}$. A 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 is any function $x \in \mathbb{R}^{n_0} \mapsto z^{(L+1)}(x) \in \mathbb{R}^{n_{L+1}}$ of the form*

$$z^{(L+1)}(x) = A^{(L+1)} \circ \sigma \circ A^{(L)} \circ \cdots \circ \sigma \circ A^{(1)}(x) \tag{1.1}$$

where $A^{(\ell)} : \mathbb{R}^{n_{\ell-1}} \to \mathbb{R}^{n_\ell}$ are affine maps:

$$A^{(\ell)}(x) = W^{(\ell)}x + b^{(\ell)}, \qquad W^{(\ell)} \in \mathbb{R}^{n_\ell \times n_{\ell-1}},\, b^{(\ell)} \in \mathbb{R}^{n_\ell}$$

and σ applied to a vector is shorthand for σ applied to each component.

The depth L and widths n_ℓ constitute the *network architecture*, whereas the entries $W_{ij}^{(\ell)}$ and components $b_i^{(\ell)}$ are the *network parameters*, respectively called weights and biases. Note that in components, the equation (1.1) reads

$$z_i^{(\ell+1)}(x) = \begin{cases} b_i^{(\ell+1)} + \sum_{j=1}^{n_\ell} W_{ij}^{(\ell+1)} \sigma\left(z_j^{(\ell)}(x)\right), & \ell \geq 1 \\ b_i^{(1)} + \sum_{j=1}^{n_1} W_{ij}^{(1)} x_j, & \ell = 0 \end{cases}. \tag{1.2}$$

Definition 1.2. *Fix $L, n_0, \dots, n_{L+1} \geq 1$ and $\sigma : \mathbb{R} \to \mathbb{R}$. A random depth L neural network with input dimension n_0, hidden layer widths $n_1, \dots, n_L$, output dimension n_{L+1}, and non-linearity σ is obtained by taking random weigths and biases in* (1.2)*:*

$$W_{ij}^{(\ell+1)} \sim \mathcal{N}\left(0, \frac{C_W}{n_\ell}\right), \quad b_i^{(\ell+1)} \sim \mathcal{N}(0, C_b) \qquad \textit{independent.} \tag{1.3}$$

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:

$$\sigma(t) = \mathrm{ReLU}(t) := t\mathbf{1}_{t \geq 0}.$$

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. We consider all one hidden layer ReLU networks with input and output dimensions equal to 1 (i.e. $L = 1, n_0 = n_2 = 1$):

$$z(x) = b^{(2)} + \sum_{i=1}^{n_1} W_i^{(2)} \sigma\left(W_i^{(1)} x + b_i^{(1)}\right), \qquad b_i^{(1)}, W_i^{(1)}, W_i^{(2)}, b^{(2)} \in \mathbb{R}. \tag{1.4}$$

The key point is the following

Lemma 1.3. *The space of functions obtained by varying $W_i^{(1)}, b_i^{(1)}, W_i^{(2)}, b^{(2)}$ in (1.4) 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. □

1.3 Typical Use of Neural Networks

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. The usual learning process can be roughly divided into four steps:

1. **Data Acquisition.** Obtain a dataset $\mathcal{D} = \{(x_k, y_k))\}_{k=1}^K, \quad y_k = f(x_k) + \text{noise}.$
2. **Architecture Selection.** Choose an architecture $L, n_1, \ldots, n_L, \sigma$, which specifies the form of the computation $x \mapsto z^{(L+1)}(x)$ but does not determine the trainable parameters $\theta = \left\{W^{(\ell)}, b^{(\ell)}\right\}$.
3. **Initialization.** Randomly select the weights and biases as in (1.3).
4. **Optimization.** Repeatedly update θ by some variant of (stochastic) gradient descent on an appropriate empirical loss, e.g.

$$\mathcal{L}_{\mathcal{D}}(\theta) = \frac{1}{|\mathcal{D}|} \sum_{(x,y)\in\mathcal{D}} \ell(y, z^{(L+1)}(x;\theta)) \tag{1.5}$$

   to obtain a final setting of parameters θ_* (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

$$\mathcal{L}_{\mathcal{D}}(\theta_*) \text{ is small} \tag{1.6}$$

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

$$\ell(y, z^{(L+1)}(x;\theta_*)) \text{ is small on average or with high probability over draws of } (x,y). \tag{1.7}$$

1.4 Broad Questions

In practice, a number of specific tricks are employed for steps $1-4$ to yield a good approximation in the sense of (1.6) and (1.7). However, that this process works well at all is somewhat surprising and motivates a number of important theoretical questions about neural networks:

Q1. **Success of non-convex optimization.** Empirically, (stochastic) gradient descent from a random initialization finds an approximate global minimum of $\mathcal{L}_{\mathcal{D}}$. However, the loss $\mathcal{L}_{\mathcal{D}}$ in (1.5) is non-convex as a function of θ. Why is this possible?

Q2. Generalization with Overparameterized Interpolation. A cornerstone of traditional statistical learning theory is the bias-variance tradeoff, which roughly says that although models with high capacity can interpolate data in the sense of (1.6), they will be prone to overfitting and hence fail to extrapolate in the sense of (1.7). Many neural networks used in practice are overparameterized:

$$\#\text{datapoints} \quad \ll \quad \#\text{trainable parameters} \tag{1.8}$$

or have seemingly high capacity with respect to some other natural complexity measure. Yet, not only are they capable of perfectly fitting the data (and even random noise) but they are also capable of extrapolating to unseen data. Why is this possible? Really there are two related questions:

- **Implicit Bias.** Under the overparameterization condition (1.8) there are infinitely many θ for which $z(x;\theta) = y$ for all $(x, y) \in \mathcal{D}$. This makes clear that which θ is chosen (and in particular which θ give rise to functions $z(x;\theta)$ that make meaningful predictions on x not in the training data) depends very much on the learning algorithm. Understanding this dictionary between learning algorithms (i.e. how to initialize and how to optimize) and the quality of learned models is the goal of the sub-field of implicit bias.
- **Benign Overfitting.** The surprising aspect of the previous discussion of implicit bias is that, among the sea of minimizers θ of the empirical loss $\mathcal{L}(\theta)$, optimizers are able to choose "good θ's." But there is another mystery. Namely, even when we only observe noise data i.e.

$$y_k = f(x_k) + \epsilon_k, \qquad \epsilon_k \sim \mathcal{N}(0, \sigma_\varepsilon^2)$$

neural networks $z(x;\theta)$ that exact fit the training data still do well in making predictions on unseen inputs. This seems to flies in the face of the traditional imperative to regularize models and is known as benign overfitting.

Q3. **Feature/Transfer Learning.** Perhaps the most well-understood and versatile forms of learning rely 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 $\Phi(x) = (\phi_j(x),\, j \geq 1)$ is a given set of (usually linearly independent) 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. In contrast, the components of the output of a neural network can be written as

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

Hence, neural network training consists of learning both the basis 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. However, precious few precise mathematical characterizations of what kinds of bases are learnable are known.

Q4. Scaling Laws. Neural networks are models with many large parameters e.g.

- network depth L
- network width n
- number of training datapoints P
- input dimension n_0
- compute budget C.

A key question is in which combinations these parameters affect the quality of trained networks. Empirically, if we plot say ℓ_2 generalization error against number of network parameters, or compute budget, or number of training data points we get a power law behavior. Such observations generically go under the name of "scaling laws" for neural networks. For example, we'll see below that the properties of the distribution of random neural networks depends primarily on the ratio L/n of depth-to-width rather than depth or width individually.

1.5 Goal for Lectures 2, 3

The overall goal for lectures 2,3 is to understand the law of the field

$$x \in \mathbb{R}^{n_0} \;\mapsto\; z^{(L+1)}(x) \in \mathbb{R}^{n_{L+1}}$$

given by a random neural network (see Definition 1.2) when the hidden layer widths are large:

$$n_1, \ldots, n_L \simeq n \gg 1.$$

The key points will be

- For any fixed L, when $n \to \infty$, the fields $x \mapsto z^{(L+1)}(x)$ converge to Gaussian processes with mean 0 and a covariance function $K^{(L+1)}$ determined by C_b, C_W, σ.
- We'll see a principled way of choosing C_b, C_W (called tuning to criticality), for which $K^{(\ell)}$ are well-behaved at large ℓ
- We'll see that at finite but large width the statistics of $z^{(L+1)}(x)$ are determined by
  - the universality class of the non-linearity σ (i.e. large ℓ behavior of $K^{(\ell)}$)

– the effective depth (or effective complexity)

$$\frac{1}{n_1} + \cdots + \frac{1}{n_L} \simeq \frac{L}{n}.$$

Specifically, at init, L/n measures both correlations between neurons and fluctuations in both values and gradients. This suggests an interesting phase diagram (see Figure 1).

2 Infinite Width Networks at Fixed Depth: GP/NTK Regime

In the first part of this lecture we study the Gaussian process limit of wide neural networks at initialization. The setup is recalled in §2.1. The main result, Theorem 2.1, shows that in this regime random neural networks converge to Gaussian processes.[1] The proof is given in §2.3 and §2.4. Having understood the initialization behavior of networks at finite depth and infinite width, we then turn to studying optimization 2.5. The main result is Theorem 2.4, which shows that networks at finite depth and infinite width trained by gradient flow on an empirical MSE loss are equivalent to kernel methods. We indicate the main ideas in §2.5.1 and §2.5.2.

2.1 Random Neural Networks

Specifically, we consider a random fully connected neural network with input dimension n_0, L hidden layers of widths $n_1, \ldots, n_L$, output dimension n_{L+1}, and non-linearity σ. Recall from Definition 1.2 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 $z^{(\ell)}(x)$

$$z_i^{(\ell+1)}(x) = \begin{cases} b_i^{(\ell+1)} + \sum_{j=1}^{n_\ell} W_{ij}^{(\ell+1)} \sigma\left(z_j^{(\ell)}(x)\right), & \ell \geq 1 \\ b_i^{(1)} + \sum_{j=1}^{n_1} W_{ij}^{(1)} x_j, & \ell = 0 \end{cases}$$

with

$$W_{ij}^{(\ell+1)} \sim \mathcal{N}\left(0, \frac{C_W}{n_\ell}\right), \quad b_i^{(\ell+1)} \sim \mathcal{N}(0, C_b) \qquad \text{independent.}$$

2.2 Statement of GP Limit at Infinite Width and Finite Depth

In this lecture we will prove two concrete results about the forward pass $x \mapsto z^{(L+1)}(x)$. The first result pertains to the limit when $n_1, \ldots, n_L \to \infty$. To state it, 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 2.1. *Fix L, σ, n_0, n_{L+1} and a polynomially bounded $\sigma : \mathbb{R} \to \mathbb{R}$. Then the sequence of fields $x \mapsto z^{(L+1)}(x)$ converges in distribution to a mean zero Gaussian process with iid components:*

$$\lim_{n_1,\ldots,n_L \to \infty} z^{(L+1)}(\cdot) \stackrel{d}{=} \mathcal{GP}(0, K^{(L+1)})^{\otimes n_{L+1}},$$

where

$$\lim_{n_1,\ldots,n_L \to \infty} \mathrm{Cov}\left(z_i^{(\ell+1)}(x_\alpha), z_j^{(\ell+1)}(x_\beta)\right) = \delta_{ij} K^{(L+1)}(x_\alpha, x_\beta),$$

with

$$K^{(\ell+1)}(x_\alpha, x_\beta) = C_b + C_W \mathbb{E}_{K^{(\ell)}}\left[\sigma\left(z_1^{(\ell)}(x_\alpha)\right) \sigma\left(z_1^{(\ell)}(x_\beta)\right)\right]. \tag{2.1}$$

[1] An important caveat is that we consider random neural networks with the standard NTK initialization give in (1.3). Other initialization schemes are quite interesting and lead to very different behaviors. See e.g. [MMN18, RVE18, YH21, SS20, SS21, YS20].

Remark 2.2. *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 [Han21b] for a more general statement for fully connected networks.*

We outline the proof Theorem 2.1 in §2.4.

2.3 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

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

Moreover, these conditional covariances are examples of *collective observables*

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

which play the role of order parameters. The key technical result, which I will take for granted in these notes is the following:

Theorem 2.3 (Structure Theorem for Collective Observables)**.** *Suppose $\sigma : \mathbb{R} \to \mathbb{R}$ is polynomially bounded. For $k \geq 1$ suppose that $f_1, \dots, f_k : \mathbb{R}^{|A|} \to \mathbb{R}$ are polynomially bounded. Then*

$$\mathbb{E}\left[\prod_{j=1}^{k}\left(\mathcal{O}_{f_j}^{(\ell)} - \mathbb{E}\left[\mathcal{O}_{f_j}^{(\ell)}\right]\right)\right] = O\left(n^{-\lceil \frac{k}{2} \rceil}\right), \tag{2.2}$$

where

$$n := \min\left\{n_1, \dots, n_\ell\right\}.$$

Proof. See Theorem 3.1 and Lemma 7.5 in [Han21a]. □

2.4 Derivation of GP Limit: Proof of Theorem 2.1

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

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

we have

$$\lim_{n_1,\ldots,n_L\to\infty} \mathbb{E}\left[\exp\left\{-i\sum_{m=1}^{n_{L+1}} z_m^{(L+1)}(x_A)\cdot\xi_m\right\}\right] = \exp\left\{-\frac{1}{2}\sum_{m=1}^{n_{L+1}} \xi_m^T K_A^{(L+1)}\xi_m\right\} \tag{2.3}$$

with

$$K_A^{(L+1)} = \left(K^{(L+1)}(x_\alpha, x_\beta)\right)_{\alpha,\beta\in A}$$

satisfying (2.1). To establish (2.3) note that for any $\ell = 1,\ldots,L$ recall that

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

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

$$\begin{aligned}\mathbb{E}\left[\exp\left\{-i\sum_{m=1}^{n_{L+1}} z_m^{(L+1)}(x_A)\cdot\xi_m\right\}\right] &= \mathbb{E}\left[\mathbb{E}\left[\exp\left\{-i\sum_{m=1}^{n_{L+1}} z_m^{(L+1)}(x_A)\cdot\xi_m\right\} \;\middle|\; z^{(L)}(x_A)\right]\right] \\ &= \mathbb{E}\left[\exp\left\{-\frac{1}{2}\sum_{m=1}^{n_{L+1}} \xi_m^T \widehat{K}_A^{(L)}\xi_m\right\}\right]. \end{aligned} \tag{2.4}$$

Theorem 2.3 guarantees that the following converges 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 (2.4) 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

$$\begin{aligned}\lim_{n_1,\ldots,n_{\ell+1}\to\infty} \mathrm{Cov}\left(z_1^{(\ell+1)}(x_\alpha), z_1^{(\ell+1)}(x_\beta)\right) &= \lim_{n_1,\ldots,n_\ell} \mathbb{E}\left[\widehat{K}^{(\ell)}(x_\alpha, x_\beta)\right] \\ &= \lim_{n_1,\ldots,n_\ell\to\infty} \mathbb{E}\left[C_b + \frac{C_W}{n_\ell}\sum_{j=1}^{n_\ell} \sigma\left(z_j^{(\ell)}(x_\alpha)\right)\sigma\left(z_j^{(\ell)}(x_\beta)\right)\right] \\ &= C_b + C_W \mathbb{E}_{K^{(\ell)}}\left[\sigma\left(z_1^{(\ell)}(x_\alpha)\right)\sigma\left(z_1^{(\ell)}(x_\beta)\right)\right], \end{aligned}$$

which confirms (2.1). □

2.5 Optimization of Wide Networks in the NTK Regime

In this section, we present an influential line of work on the so-called NTK or linear or kernel regime of neural networks [DLT$^+$18, JGH18, LZB20]. The main result is the following

Theorem 2.4 ([DLT$^+$18, JGH18, LZB20, CB18])**.** *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. 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 (1.3). If $n_1, \ldots, n_L = \text{poly}(m)$ then with with high probability

(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

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

where

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

$$\mathcal{L}^{\text{lin}}(\theta) = \sum_{i=1}^{m} \left(z^{(L+1),\text{lin}}(x_i; \theta) = y_i \right)^2$$

$$z^{(L+1),\text{lin}}(x; \theta) = z^{(L+1)}(x; \theta_0) + \nabla_\theta z^{(L+1)}(x; \theta_0)\,(\theta - \theta_0)$$

2.5.1 Idea 1 in Theorem 2.4: Tangent Kernel

Consider optimizing the parameters of a model $z(x; \theta)$ by gradient descent

$$\theta_{t+1} = \theta_t - \eta \nabla_\theta \mathcal{L}(\theta_t) \tag{2.5}$$

on an empirical loss such as

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

on 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 now shows

$$z(\mathcal{D}; \theta_{t+1}) = z(\mathcal{D}; \theta_t) - \eta\ \Theta(\theta_t) \nabla_z \mathcal{L}(z(\mathcal{D})) + O(\eta^2),$$

where

$$\Theta(\theta) := \left(\Theta(x_i, x_j; \theta)\right)_{i,j=1,\ldots,m}, \qquad Y = (y_1, \ldots, y_m),$$

where $\Theta(\theta)$ is the so-called tangent kernel:

Definition 2.5 (Tangent kernel)**.** *Let $x \in \mathbb{R}^{n_{in}} \mapsto z(x;\theta) \in \mathbb{R}$ be a parameterized family of functions with $\theta = (\theta_\mu,\, \mu = 1, \ldots, \#parameters)$. The parameter Jacobian of this model is*

$$J(\theta) \in \mathbb{R}^{n_{in}}, \qquad J(x;\theta) := \nabla_\theta z(x;\theta).$$

The tangent kernel of this model is

$$\Theta : \mathbb{R}^{n_{in}} \times \mathbb{R}^{n_{in}} \to \mathbb{R}, \qquad \Theta(x,x') := J(x;\theta)^T J(x';\theta),$$

or more explicitly:

$$\Theta(x,x';\theta) := \sum_{\mu=1}^{\#parameters} \partial_{\theta_\mu} z(x;\theta) \partial_{\theta_\mu} z(x';\theta) \tag{2.6}$$

Thus, $\Theta(\theta_t)^{-1}$ plays the role of a Riemannian metric at the function $z(\cdot;\theta_t)$ that is used to perform gradient descent in these new coordinates. We therefore have the following simple but fundamental result

Lemma 2.6. *Suppose there exists $\delta > 0$ so that we have the following inequality of PSD matrices*

$$\Theta(\theta_t) \geq \delta I \qquad \forall t \geq 0 \tag{2.7}$$

and 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

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

for some $c > 0$.

Proof. The condition (2.7) ensures that

$$\langle \Theta(\theta_t) \nabla_z \mathcal{L}(z(\mathcal{D})), \nabla_z \mathcal{L}(z(\mathcal{D};\theta_t)) \rangle \geq c \, ||\nabla_z \mathcal{L}(z(\mathcal{D};\theta_t))||^2 .$$

Since $\mathcal{L}$ is strictly convex, this guarantees exponential convergence to a minimum. □

Corollary 2.7. *Suppose*

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

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

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

2.5.2 Idea 2 in Theorem 2.4: Linearization at Large Width

The basic idea in the NTK literature for establishing (2.7) is

(i) Show that the desired estimate

$$\Theta(\theta_0) \geq \delta I$$

holds with high probability at initialization using matrix concentration.

(ii) Show that

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

using

$$||\Theta(\theta_0) - \Theta(\theta_t)|| \leq \int_0^t ||\text{Hess}_\theta z(\theta_s)||\, ||\nabla \mathcal{L}(\theta_s)||\, ds$$

and

$$||\text{Hess}_\theta z(\theta)|| \leq \frac{\text{poly}\,(\#\text{data})}{\sqrt{\text{width}}}, \qquad \forall\, ||\theta - \theta_0|| \leq R$$

Let's see how this works in the particularly simple case of one layer networks with NTK parameterization:

$$z^{(2)}(x;\theta) = \sum_{i=1}^{n_1} \frac{1}{\sqrt{n_1}} W_i^{(2)} \sigma\left(W_i^{(1)} \frac{x}{\sqrt{n_0}}\right).$$

We've turned off the biases for simplicity but the computations below work with minimal changes either way. First note the following

$$\frac{\partial}{\partial W_i^{(1)}} z^{(2)}(x;\theta) = \frac{1}{\sqrt{n_1}} W_i^{(2)} \sigma'\left(W_i^{(1)} \frac{x}{\sqrt{n_0}}\right) \frac{x}{\sqrt{n_0}}$$
$$\frac{\partial}{\partial W_i^{(2)}} z^{(2)}(x;\theta) = \frac{1}{\sqrt{n_1}} \sigma\left(W_i^{(1)} \frac{x}{\sqrt{n_0}}\right).$$

Hence, $\Theta_{\alpha\beta}(\theta)$

$$\frac{1}{n_1} \sum_{i=1}^{n_1} \left\{ \left(W_i^{(2)}\right)^2 \sigma'\left(W_i^{(1)} \frac{x_\alpha}{\sqrt{n_0}}\right) \sigma'\left(W_i^{(1)} \frac{x_\beta}{\sqrt{n_0}}\right) \frac{x_\alpha \cdot x_\beta}{n_0} + \sigma\left(W_i^{(1)} \frac{x_\alpha}{\sqrt{n_0}}\right) \sigma\left(W_i^{(1)} \frac{x_\beta}{\sqrt{n_0}}\right) \right\},$$

making $\Theta(\theta_0)$ an average of iid order 1 random matrices. On the other hand

$$\frac{\partial^2}{\partial W_i^{(*_1)} \partial W_j^{(*_2)}} z^{(2)}(x;\theta) = \frac{1}{\sqrt{n_1}} \delta_{ij}\, (\text{order } 1), \qquad *_1, *_2 \in \{1,2\}.$$

Hence,

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

Some simple perturbation theory shows that

$$||\theta - \theta_0|| \text{ not too large} \qquad \Longrightarrow \qquad \left|\left|\text{Hess}_\theta z^{(2)}(x;\theta)\right|\right| = O\left(\frac{1}{\sqrt{n_1}}\right).$$

3 The Importance of Finite Width Effects at Large Depth

In the previous lecture we studied neural networks at a fixed depth and infinite width. Theorem 2.1 says that their outputs are Gaussian Processes when their parameters are initialized in a standard way. Further, Theorem 2.4 says that, with small learning rates and fixed dataset sizes, they become linear models when their parameters are optimized to fit a loss such a mean squared error. This is both illuminating and deeply disappointing.

Indeed, the linearization result (statement (ii) in Theorem 2.4) shows that, at infinite width, neural networks with a fixed number of hidden layers cannot learn data-dependent features in the sense that they simply perform regression in the RKHS determined by the neural tangent kernel $\Theta(\theta_0)$ at initialization. Despite this, one of the most important properties of neural networks in practice is their ability to learn data-dependent features useful for transfer learning (see **Q3** in §1.4). To study neural network models in capable of feature learning several approaches are possible:

- Studying networks at finite width [Han18, HN19, RYH22, Han21a]
- Studying networks at large learning rates [LBD$^+$20, ZLRB22]
- Studying optimization in the regime of simultaneously growing number of datapoints and network width [APP$^+$22, LS21, SR21, NR21]
- Studying networks with mean-field initialization [YH21, MMN18, SS20, SS21, YS20, RVE18]

In this lecture, we will explore the first approach. Will find the following

Message: Both the extent of feature learning during training and the deviation from Gaussianity at initialization are controlled by the ratio of depth to width.

This point has been taken up in a range of prior works [Han18, HR18, HN19, HN20, Han21a, RYH22, Yai20].

Our goal here is to give a glimpse into how such results might be derived. To start let's consider the recursion relation (2.1), which describes the infinite width Gaussian Process covariance $K^{(\ell+1)}$ in terms of C_b, C_W, σ and $K^{(\ell)}$. As shown most fully in [Han21a] 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

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

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 3.1. *Fix L, n_0, n_{L+1}, σ. Suppose that the weights and biases are chosen as in* (1.3) *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:

$$\kappa_4^{(\ell+1)}(x) = \frac{C_W^2}{n_\ell}\mathrm{Var}_{K^{(\ell)}}\left[\sigma^2\right] + \left(\chi_{||}^{(\ell)}(K^{(\ell)}(x,x))\right)^2 \kappa_4^{(\ell)}(x) + O(n^{-2}). \tag{3.1}$$

Thus, at criticality and uniform width ($n_\ell = n$), 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

$$\begin{aligned}
&\mathbb{E}\left[f\left(z_i^{(\ell)}(x),\, 1 \leq i \leq m\right)\right] \\
&\quad = \mathbb{E}_{G^{(\ell)}}\left[f\left(z_i^{(\ell)}(x),\, 1 \leq i \leq m\right)\right] \\
&\quad + \frac{\kappa_4^{(\ell+1)}(x)}{8}\mathbb{E}_{K^{(\ell)}}\left[\left(\sum_{j=1}^m \partial_{z_j}^4 + \sum_{\substack{j_1,j_2=1 \\ j_1\neq j_2}}^m \partial_{z_{j_1}}^2 \partial_{z_{j_2}}^2\right) f\left(z_i^{(\ell)}(x),\, 1 \leq i \leq m\right)\right] \\
&\quad + O(n^{-2}).
\end{aligned} \tag{3.2}$$

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 [Han21a]. We give the sketch of proof in §3.2.

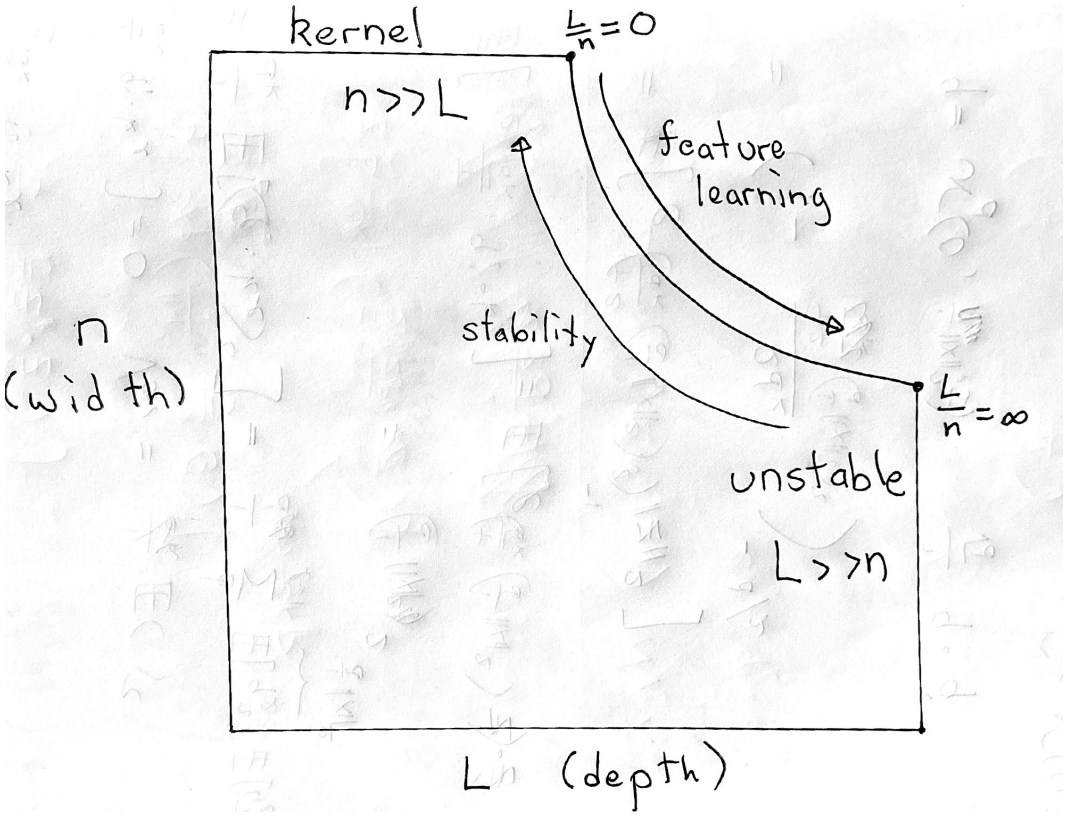

Figure 1: Partial Phase Diagram for Fully Connected Networks with NTK Initialization

3.1 Intuition for Appearance of L/n

Before proceeding to the study of non-linear networks, we pause to explain a simple intuition for why it is L/n, rather than some other combination of L and n, that should appear. For this, let us consider the very simple case of random matrix products

$$n_\ell \equiv n,\ \sigma(t) = t,\ C_b = 0, C_W = 1$$

so that

$$z^{(L+1)}(x) = W^{(L+1)} \cdots W^{(1)} x, \qquad W_{ij}^{(\ell)} \sim \mathcal{N}(0, 1/n)\ iid.$$

Assuming for convenience that $||x|| = 1$, let's try to understand what is perhaps the simplest random variable

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

associated to our matrix product. 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

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

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

$$Wu \stackrel{d}{=} \mathcal{N}\left(0, \frac{1}{n}\mathrm{I}_n\right), \qquad ||Wu||^2 \stackrel{d}{=} \frac{1}{n}\chi^2_n, \qquad ||Wu|| \perp \frac{Wu}{||Wu||},$$

where $\perp$ denotes independence. To use this let's write

$$X_{n,L+1} = \left|\left|W^{(L+1)}\cdots W^{(1)}x\right|\right| = \left|\left|W^{(L+1)}\cdots W^{(2)}\frac{W^{(1)}x}{||W^{(1)}x||}\right|\right| \left|\left|W^{(1)}x\right|\right|.$$

Note that

$$\left(\frac{1}{n}\chi^2_n\right)^{1/2} \stackrel{d}{=} \left|\left|W^{(1)}x\right|\right|, \qquad \left|\left|W^{(1)}x\right|\right| \perp \frac{W^{(1)}x}{||W^{(1)}x||} \in S^{n-1}.$$

Thus, in fact the presentation above allows us to write $X_{n,L+1}$ as a product of two independent terms! Proceeding in this way, we obtain the following equality in distribution:

$$X_{n,L+1} \stackrel{d}{=} \exp\left[\sum_{\ell=1}^{L+1} Y_\ell\right], \qquad Y_\ell \sim \frac{1}{2}\log\left(\frac{1}{n}\chi^2_n\right) \ iid.$$

Exercise. Show 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

$$X_{n,L+1} \stackrel{L\gg 1}{\approx} \exp\left[\mathcal{N}\left(-\frac{L}{4n}, \frac{L}{4n}\right)\right]$$

and that **taking n large in each layer tries to make each Y_j close to 1 but with errors of size $1/n$. When we have L such errors, the total size of the error is on the order of L/n.**

3.2 Proof Sketch of Theorem 3.1

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

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

Applying Theorem 2.3 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

$$\mathbb{E}\left[f(z_i^{(\ell)},\, i=1,\ldots,m)\right] = \mathbb{E}_{G^{(\ell)}}\left[f\right]+\frac{1}{8}\mathbb{E}\left[\left(\Delta^{(\ell-1)}\right)^2\right]\mathbb{E}_{G^{(\ell)}}\left[\left(\sum_{j=1}^m \partial^2_{z_j^{(\ell)}}\right)^2 f\right]+O(n^{-2}). \tag{3.3}$$

Exercise. Show that

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

This proves (3.2). 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 (3.3) and some algebra completes the proof of (3.1). □

3.3 Tuning to Criticality: Analysis of GP Limit at Large Depth

We just saw in Theorem 2.1 that, at any fixed depth, random fully connected networks converges to GPs with mean 0 and covariance $K^{(L+1)}$ in the limit of infinite width. In this part of the lecture our goal is to understand how to choose C_b, C_W so that the resulting GPs kernels

$$K^{(\ell)}(x_\alpha,x_\beta) = \lim_{n_1,\ldots,n_{\ell-1}\to\infty}\mathrm{Cov}\left(z_1^{(\ell)}(x_\alpha),z_1^{(\ell)}(x_\beta)\right) = \lim_{n_1,\ldots,n_\ell}\frac{1}{n_\ell}z^{(\ell)}(x_\alpha)\cdot z^{(\ell)}(x_\beta)$$

are well-behaved (i.e. neither exponentially growing nor decaying) at large ℓ.

3.3.1 The Case of Linear Activations.

Before developing the general theory let's consider the case of linear networks in which

$$\sigma(t) = t.$$

Then (2.1) reads

$$K^{(\ell+1)}(x_\alpha, x_\beta) = C_b + C_W \mathbb{E}_{K^{(\ell)}}\left[z_1^{(\ell)}(x_\alpha) z_1^{(\ell)}(x_\beta)\right] = C_b + C_W K^{(\ell)}(x_\alpha, x_\beta).$$

Thus, a sensible choice is to set $C_b = 0, C_W = 1$. This ensures that the covariance structure of the GP is independent of ℓ. Note that if $C_W \neq 1$, then $K^{(\ell)}(x_\alpha, x_\beta)$ either grows or decays exponentially with ℓ.

3.3.2 Tuning to Criticality: Setting C_b, C_W for General Activations

The idea, which is sometimes called "tuning to criticality," is to find $K_* \geq 0$ such that

$$K_* = C_b + C_W \mathbb{E}_{K_*}\left[\sigma(z)^2\right] \tag{$*$}$$

$$\chi_{||}(K_*) := \left.\frac{\partial K^{(\ell+1)}(x_\alpha, x_\alpha)}{\partial K^{(\ell)}(x_\alpha, x_\alpha)}\right|_{K^{(\ell)}(x_\alpha, x_\alpha) = K_*} = \frac{C_W}{2}\mathbb{E}_{K_*}\left[\partial_z^2(\sigma(z)^2)\right] = 1 \tag{$||$}$$

$$\chi_{\perp}(K_*) := \left.\frac{\partial K^{(\ell+1)}(x_\alpha, x_\beta)}{\partial K^{(\ell)}(x_\alpha, x_\beta)}\right|_{K^{(\ell)}(x_\alpha, x_\alpha) = K^{(\ell)}(x_\beta, x_\beta) = K^{(\ell)}(x_\alpha, x_\beta) = K_*} = \mathbb{E}_{K_*}\left[\sigma'(z)^2\right] = 1. \tag{$\perp$}$$

3.3.3 Criticality for ReLU

The ReLU is

$$\sigma(z) = z\mathbf{1}_{z\geq 0}$$

$$K_* = C_b + C_W \mathbb{E}_{K_*}\left[z^2 \mathbf{1}_{z\geq 0}\right] = C_b + \frac{C_W}{2} K_*.$$

Note that the full recursion for $K(x_\alpha, x_\beta)$ is more complicated. Further,

$$\chi_{||}(K) = \frac{C_W}{2}\mathbb{E}_K\left[\partial_z^2 (z\mathbf{1}_{z\geq 0})^2\right] = C_W \mathbb{E}_K\left[\mathbf{1}_{z\geq 0}\right] = \frac{C_W}{2}$$
$$\chi_{\perp}(K) = C_W \mathbb{E}_K\left[\mathbf{1}_{z\geq 0}\right] = \frac{C_W}{2}.$$

Thus, we find that criticality is achieved by setting $C_b = 0, C_W = 2$.

3.4 Extensions and Open Problems

We conclude with a list of open problems:

1. Obtain the precise rate of convergence as a function of depth (say in the Wasserstein or some other metric) between $z^{(L+1)}(x)$ and the limiting GP, after tuning to criticality.

2. Obtain the finite width corrections to the joint distribution *across layers* of components of $z^{(\ell)}(x)$.

3. Find double-scaling limit of $n, L \to \infty$ so that not just the marginal distribution of $z_i^{(\ell)}(x)$ evaluated at a single point but also the joint over many x's is non-degenerate. This requires "shaping" of the neural network activations. See e.g. [LNR22].

4. Compute the Lipschitz constant for a random ReLU network. Applications to adversarial examples [BBC21, BCGTdC21, MW22, DS20].

5. Compute the depth-dependence of the results in the mean-field papers such as [MMN18, RVE18, YH21, SS20, SS21, YS20].

References

[APP+22] S Ariosto, R Pacelli, M Pastore, F Ginelli, M Gherardi, and P Rotondo. Statistical mechanics of deep learning beyond the infinite-width limit. *arXiv preprint arXiv:2209.04882*, 2022.

[BBC21] Peter Bartlett, Sébastien Bubeck, and Yeshwanth Cherapanamjeri. Adversarial examples in multi-layer random relu networks. *Advances in Neural Information Processing Systems*, 34:9241–9252, 2021.

[BCGTdC21] Sébastien Bubeck, Yeshwanth Cherapanamjeri, Gauthier Gidel, and Remi Tachet des Combes. A single gradient step finds adversarial examples on random two-layers neural networks. *Advances in Neural Information Processing Systems*, 34:10081–10091, 2021.

[CB18] Lenaic Chizat and Francis Bach. A note on lazy training in supervised differentiable programming. *arXiv preprint arXiv:1812.07956*, 2018.

[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.

[DS20] Amit Daniely and Hadas Shacham. Most relu networks suffer from l2 adversarial perturbations. *Advances in Neural Information Processing Systems*, 33:6629–6636, 2020.

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

[Han21a] Boris Hanin. Correlation functions in random fully connected neural networks at finite width. *arXiv preprint arXiv:2204.01058*, 2021.

[Han21b] Boris Hanin. Random neural networks in the infinite width limit as gaussian processes. *arXiv preprint arXiv:2107.01562*, 2021.

[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.

[HN19] Boris Hanin and Mihai Nica. Finite depth and width corrections to the neural tangent kernel. *ICLR 2020 and arXiv:1909.05989*, 2019.

[HN20] 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.

[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.

[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.

[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.

[LNR22] Mufan Bill Li, Mihai Nica, and Daniel M Roy. The neural covariance sde: Shaped infinite depth-and-width networks at initialization. *arXiv preprint arXiv:2206.02768*, 2022.

[LS21] Qianyi Li and Haim Sompolinsky. Statistical mechanics of deep linear neural networks: The backpropagating kernel renormalization. *Physical Review X*, 11(3):031059, 2021.

[LZB20] Chaoyue Liu, Libin Zhu, and Mikhail Belkin. Toward a theory of optimization for over-parameterized systems of non-linear equations: the lessons of deep learning. *arXiv preprint arXiv:2003.00307*, 2020.

[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.

[MW22] Andrea Montanari and Yuchen Wu. Adversarial examples in random neural networks with general activations. *arXiv preprint arXiv:2203.17209*, 2022.

[NR21] Gadi Naveh and Zohar Ringel. A self consistent theory of gaussian processes captures feature learning effects in finite cnns. *Advances in Neural Information Processing Systems*, 34, 2021.

[RVE18] Grant M Rotskoff and Eric Vanden-Eijnden. Neural networks as interacting particle systems: Asymptotic convexity of the loss landscape and universal scaling of the approximation error. *arXiv preprint arXiv:1805.00915*, 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.

[SR21] Inbar Seroussi and Zohar Ringel. Separation of scales and a thermodynamic description of feature learning in some cnns. *arXiv preprint arXiv:2112.15383*, 2021.

[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.

[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.

[YS20] Jiahui Yu and Konstantinos Spiliopoulos. Normalization effects on shallow neural networks and related asymptotic expansions. *arXiv preprint arXiv:2011.10487*, 2020.

[ZLRB22] Libin Zhu, Chaoyue Liu, Adityanarayanan Radhakrishnan, and Mikhail Belkin. Quadratic models for understanding neural network dynamics. *arXiv preprint arXiv:2205.11787*, 2022.