Lecture 11: Moving Beyond the Linear Approximation Regime

Geoff Pleiss

Thus far we have examined neural networks through the lens of linearization. While we have shown that this approximation is (under certain conditions) exact for infinitely wide networks, there are reasons to believe that the linear approximation is not the whole story, especially for real-world neural networks.

1. If the Jacobian of the neural network stays fixed during training, then the neural network is not performing **"feature learning."** The story of neural networks (and what is seen in practice) is that the hidden features learn to pick up on specific patterns in the data. (Think of face detectors for image data or word embeddings for text data.) Under the linear approximation, the features of a neural network are based solely on architecture and initialization. A neural network would thus have to have face-detecting features built in from the start rather than learning them from the data.
2. Belatedly, linearized neural networks cannot perform **transfer learning**, where the hidden representations learned on one (usually large) dataset are used to jump-start learning on a tangentially related (usually smaller) dataset. If linearized neural networks are not learning features from the data, then there is no reason to "transfer" its features to a new dataset. In practice, however, transfer learning is a powerful tool that has allowed us to achieve remarkable performance on small datasets that otherwise would not be able to be learned with neural networks.

Researchers have provided evidence supporting these thoughts:

1. Empirical studies have demonstrated that the Jacobian of the neural network deviates from its initialization Fort et al. [2020].
2. Theoretical analyses have shown that neural networks do not always learn efficiently in the linearized regime. For example, Yehudai and Shamir [2019] demonstrate that it takes exponentially many samples for a linearized neural network to learn the function $f(x) = \max\{0, x\}$.
3. Other empirical and theoretical studies have shown that though SGD implicitly biases towards minimizing some functional norm, the minimized norm is often not the RKHS norm (or, more generally, some norm defined by an inner product). For example, Savarese et al. [2019] finds that the learned functions often minimize a semi-norm based on function derivatives, while Woodworth et al. [2020] suggest that the learned functions minimize a sparse functional norm.
4. Finally, if we consider wide-and-deep neural networks (e.g. networks where width and depth are scaled proportionally to one another, rather than one being fixed while the other goes to infinity), linear approximations break down Li et al. [2021].

---

1) Characterizing Feature Learning

Characterizing feature learning and deviations from linearized behaviour is an active area of research. Some researchers have aimed to examine finite-width neural networks by adding correction terms to the linearized approximation [Hanin and Nica, 2020, e.g.]. However, this approach is cumbersome and does not give much intuition into feature learning behaviour. Moreover, these results imply that feature learning relies on small width, contrary to real-world empirical results that demonstrate great empirical successes with wide neural networks [Zagoruyko and Komodakis, 2016].

Approach: scaling arguments. Instead, we take inspiration from the **scaling argument** of Chizat et al. [2019]. Recall that our optimization analysis of the NTK approximation relied on *kappa*—the change in Jacobian relative to the change in loss—being very small:

$$\kappa \asymp \frac{\left\|\nabla^2_{\boldsymbol{\theta}} f(\cdot;\boldsymbol{\theta}_0)\right\|}{\left\|\nabla_{\boldsymbol{\theta}} f(\cdot;\boldsymbol{\theta}_0)\right\|^2}.$$

For a one-hidden-layer neural network with D hidden features,

$$f(\boldsymbol{x};\boldsymbol{\theta}) = \frac{1}{\sqrt{D}}\sum_{i=1}^{D}\beta_i\phi(\boldsymbol{w}_i^\top\boldsymbol{x}), \qquad \boldsymbol{\theta} = \begin{bmatrix}\beta_1 & \dots & \beta_D & \boldsymbol{w}_1 & \dots & \boldsymbol{w}_D\end{bmatrix}^\top, \tag{1}$$

recall that the numerator scaled as $1/\sqrt{D}$ (thanks to the scaling factor) and the denominator scaled as $1/D \times D = 1$ (the square of the scaling factor time the number of terms in the vector). Thus, $\kappa \asymp 1/\sqrt{D}$, implying that we optimized while staying within the linearized approximation about the initialization.

A change in scale? What if we instead changed the scaling in Eq. (1) to $1/D$? The numerator would scale as $1/D$ (the scaling factor) and the denominator would scale as $1/D^2 \times D = 1/D$ (the scaling factor squared times the number of terms in the vector). Thus, $\kappa \asymp 1$, implying that optimizing changes the loss and the Jacobian and equal rates and that we will deviate from the linearized approximation.

We will take inspiration from this scaling argument to construct a characterization of neural networks that deviate from the linearized approximation.

---

2) The μP Regime of Linear Neural Networks

In an extremely dense paper,[1] Yang and Hu [2021] characterize the set of conditions under which neural networks with infinite width (1) optimize to a global minimum but (2) cannot be characterized by a linearization around the initialization. The so-called **maximal update parameterization**, or μP regime, occurs on the brink of stable training (hence the name) and is a set of necessary ratios to ensure feature learning even in the infinite-width limit.

In a tech report, Yang et al. [2023] introduce a more straightforward derivation of the μP regime relying on a spectral scaling argument in the same vein as Chizat et al. [2019]. We present an intuitive overview of their argument on a simplfied neural network model.

2.1 Setup: The Linear Neural Network

We will consider a three-layer neural network with *no nonlinear activation* and width-D layers:

$$f(\boldsymbol{x};\boldsymbol{\theta}) = \boldsymbol{W}_3\boldsymbol{W}_2\boldsymbol{W}_1\boldsymbol{x}, \qquad \boldsymbol{\theta} := \begin{bmatrix}\text{flatten}(\boldsymbol{W}_1) & \text{flatten}(\boldsymbol{W}_2) & \text{flatten}(\boldsymbol{W}_3)\end{bmatrix},$$

where $\boldsymbol{W}_1, \boldsymbol{W}_2, \boldsymbol{W}_3$ are learnable parameters. If $f : \mathbb{R}^P \to \mathbb{R}$, then:

$$\boldsymbol{W}_1 \in \mathbb{R}^{D\times P}, \quad \boldsymbol{W}_2 \in \mathbb{R}^{D\times D}, \quad \boldsymbol{W}_3 \in \mathbb{R}^{1\times D}.$$

We further assume that the entries of $\boldsymbol{W}_i$ are initialized i.i.d. from $\mathcal{N}(0, \sigma_i^2)$.

[1] This paper is technically part 4 of a much longer manuscript [Yang, 2019] introducing the **Tensor Program** framework. This framework can be used to (1) derive limiting kernels of any architecture and (2) characterize when the linearization approximation holds or breaks down. However, the original manuscript was so dense that Greg Yang broke it up into four sections, each published as standalone papers.

We will train f on a *single data point* $\boldsymbol{x}, y$. Denote $\boldsymbol{h}_i$ as the "features" or "activations" of $\boldsymbol{x}$ in layer i:

$$\boldsymbol{h}_i = \boldsymbol{W}_i \boldsymbol{h}_{i-1}, \qquad \boldsymbol{h}_0 = \boldsymbol{x}, \quad \boldsymbol{h}_3 = f(\boldsymbol{x}; \boldsymbol{\theta}),$$

with $\boldsymbol{h}_0 \in \mathbb{R}^P$, $\boldsymbol{h}_1 \in \mathbb{R}^D$, $\boldsymbol{h}_2 \in \mathbb{R}^D$, and $\boldsymbol{h}_3 \in \mathbb{R}$.

We will observe what happens to $\boldsymbol{h}_1, \boldsymbol{h}_2, \boldsymbol{h}_3$ as well as $\boldsymbol{W}_1, \boldsymbol{W}_2, \boldsymbol{W}_3$ after one gradient step of training with learning rate η on the loss $\mathcal{L}(\boldsymbol{W}_1, \boldsymbol{W}_2, \boldsymbol{W}_3) := \mathcal{L}$. The changes to $\boldsymbol{W}_1, \boldsymbol{W}_2, \boldsymbol{W}_3$, denoted $\Delta\boldsymbol{W}_1, \Delta\boldsymbol{W}_2, \Delta\boldsymbol{W}_3$, are given by the gradient descent update with layer-wise learning rates η_1, η_2, η_3:

$$\Delta\boldsymbol{W}_1 = -\eta_1 \nabla_{\boldsymbol{W}_1} \mathcal{L}, \qquad \Delta\boldsymbol{W}_2 = -\eta_2 \nabla_{\boldsymbol{W}_2} \mathcal{L}, \qquad \Delta\boldsymbol{W}_3 = -\eta_3 \nabla_{\boldsymbol{W}_3} \mathcal{L}.$$

We will also denote $\Delta\boldsymbol{h}_1, \Delta\boldsymbol{h}_2, \Delta\boldsymbol{h}_3$ as the changes to the features $\boldsymbol{h}_1, \boldsymbol{h}_2, \boldsymbol{h}_3$ after one gradient step:

$$\begin{aligned} \Delta\boldsymbol{h}_i &= (\boldsymbol{W}_i + \Delta\boldsymbol{W}_i)(\boldsymbol{h}_{i-1} + \Delta\boldsymbol{h}_{i-1}) - \boldsymbol{h}_i \\ &= (\Delta\boldsymbol{W}_i \boldsymbol{h}_{i-1}) + (\boldsymbol{W}_i \Delta\boldsymbol{h}_{i-1}) + (\Delta\boldsymbol{W}_i \Delta\boldsymbol{h}_{i-1}) \end{aligned} \tag{2}$$

If the properties of $\Delta\boldsymbol{W}_1, \Delta\boldsymbol{W}_2, \Delta\boldsymbol{W}_3$ and $\Delta\boldsymbol{h}_1, \Delta\boldsymbol{h}_2, \Delta\boldsymbol{h}_3$ are "stable" then they will hold throughout training and thus will determine the network's training dynamics.

Our goal is to characterize the dynamics of $\Delta\boldsymbol{W}_1, \Delta\boldsymbol{W}_2, \Delta\boldsymbol{W}_3$ and $\Delta\boldsymbol{h}_1, \Delta\boldsymbol{h}_2, \Delta\boldsymbol{h}_3$ in terms of

1. the initialization variance $\sigma_1^2, \sigma_2^2, \sigma_3^2$ and
2. the learning rates η_1, η_2, η_3.

2.2 Conditions for Stable Training

What is required for training to be "stable?" Yang et al. [2023] propose two conditions:

1. (**Equi-Scale condition**): as is the norm for neural networks, each hidden feature $[h_i]_j$ should roughly be on the same scale, and thus:

$$\|\boldsymbol{h}_1\|_2, \|\boldsymbol{h}_2\|_2 = \Theta\left(\sqrt{D}\right), \qquad \|\boldsymbol{h}_3\|_2 = |h_3| = \Theta\left(1\right).$$

2. (**Feature learning condition**): each hidden feature $[h_i]_j$ should have meaningful but non-divergent updates, and thus:

$$\|\Delta\boldsymbol{h}_1\|_2, \|\Delta\boldsymbol{h}_2\|_2 = \Theta\left(\sqrt{D}\right), \qquad \|\Delta\boldsymbol{h}_3\|_2 = |\Delta h_3| = \Theta\left(1\right).$$

Let's consider what happens if each of these conditions does not hold:

- The first condition stipulates that each hidden feature has a non-negligible mass. It is challenging to design a neural network where this condition does not hold; most neural networks hidden layers are built using components that roughly preserve the scale of the input.
- The second condition stipulates that each hidden features also have a $\Theta(1)$ update during training. If the updates are $\omega(1)$ then the hidden features would diverge. If the update are $o(1)$ then the hidden features remain constant and no "feature learning" occurs.

2.3 Sufficient Spectral Conditions

In general, it is challenging to enforce these properties on the hidden activations themselves. However, a sufficient set of conditions to ensure these hidden activation properties are the following *spectral constraints on the weight matrices*:

$$\|\boldsymbol{W}_1\| = \Theta\left(\sqrt{D/P}\right), \qquad \|\boldsymbol{W}_2\| = O\left(1\right), \qquad \|\boldsymbol{W}_3\| = O\left(1/\sqrt{D}\right).$$
$$\|\Delta\boldsymbol{W}_1\| = \Theta\left(\sqrt{D/P}\right), \qquad \|\Delta\boldsymbol{W}_2\| = O\left(1\right), \qquad \|\Delta\boldsymbol{W}_3\| = O\left(1/\sqrt{D}\right).$$

where $\|\boldsymbol{W}\|$ is the spectral norm of the matrix $\boldsymbol{W}$, i.e. the largest singular value of $\boldsymbol{W}$. To see why these these conditions are sufficient, we can apply submultiplicativity of the spectral norm:

$$\|\boldsymbol{h}_1\|_2 \leq \|\boldsymbol{W}_1\|\|\boldsymbol{x}\|_2 = \Theta\left(\sqrt{D}\right),$$
$$\|\boldsymbol{h}_2\|_2 \leq \|\boldsymbol{W}_2\|\|\boldsymbol{h}_1\|_2 = \Theta\left(\sqrt{D}\right),$$
$$\|\boldsymbol{h}_3\|_2 \leq \|\boldsymbol{W}_2\|\|\boldsymbol{h}_1\|_2 = \Theta\left(1\right),$$

and, by applying the same bound likewise to Eq. (2), we have

$$\|\Delta\boldsymbol{h}_1\|_2 \leq \|\Delta\boldsymbol{W}_1\|\|\boldsymbol{x}\|_2 = \Theta\left(\sqrt{D}\right),$$
$$\|\Delta\boldsymbol{h}_2\|_2 \leq \|\Delta\boldsymbol{W}_2\|\|\boldsymbol{h}_1\|_2 + \|\boldsymbol{W}_2\|\|\Delta\boldsymbol{h}_1\|_2 + \|\Delta\boldsymbol{W}_2\|\|\Delta\boldsymbol{h}_1\|_2 = \Theta\left(\sqrt{D}\right),$$
$$\|\Delta\boldsymbol{h}_3\|_2 \leq \|\Delta\boldsymbol{W}_3\|\|\boldsymbol{h}_2\|_2 + \|\boldsymbol{W}_3\|\|\Delta\boldsymbol{h}_2\|_2 + \|\Delta\boldsymbol{W}_3\|\|\Delta\boldsymbol{h}_2\|_2 = \Theta\left(1\right).$$

It turns out that these submultiplicative bounds are tight. Recall from Lecture 4 that the singular values of a matrix $\boldsymbol{W} \in \mathbb{R}^{M\times m}$ with i.i.d. $\mathcal{N}(0,\sigma^2)$ entries[2] adhere to a (scaled) Marchenko-Pastur distribution as $M, m \to \infty$ simultaneously. The largest singular value supported by the Marchenko-Pastur distribution is $\sigma(\sqrt{M}+\sqrt{m})$, and thus we expect the spectral norm of $\boldsymbol{W}$ to concentrate around this value. For $\boldsymbol{W}_1$, we have $M = D$ and $m = P$, and thus

$$\|\boldsymbol{W}_1\| \approx \sigma_1(\sqrt{D}+\sqrt{P}) = \Theta\left(\sigma_1\sqrt{D}\right).$$

Moreover, by the linear Gaussian identity, the entries of $\boldsymbol{W}_1\boldsymbol{x}$ are i.i.d. $\mathcal{N}(0,\sigma_1^2\|\boldsymbol{x}\|_2^2)$. Applying any standard concentration inequality, we see that the norm of $\boldsymbol{h}_{i+1} \in \mathbb{R}^D$ concentrates around $\sqrt{D}\sigma_i\|\boldsymbol{x}\|_2$, and thus $\|\boldsymbol{h}_1\|_2 = \Theta(\|\boldsymbol{W}_1\|\|\boldsymbol{h}_0\|_2)$. Analogous results hold for $\|\boldsymbol{h}_2\|_2$ and $\|\boldsymbol{h}_3\|_2$.

Now consider the bound on $\|\Delta\boldsymbol{h}_2\|$. By backpropagation we have that

$$\Delta\boldsymbol{W}_1 = -\eta_1\nabla_{\boldsymbol{W}_1}\mathcal{L} = -\eta_1\nabla_{\boldsymbol{h}_1}\mathcal{L}\boldsymbol{x}^\top,$$

and so the update $\Delta\boldsymbol{W}_1$ is rank-one and aligned with $\boldsymbol{h}_1$:

$$\|\Delta\boldsymbol{h}_1\|_2 = \|\Delta\boldsymbol{W}_1\boldsymbol{x}\|_2 = \eta_1\|\nabla_{\boldsymbol{h}_1}\mathcal{L}\boldsymbol{x}^\top\boldsymbol{x}\|_2 = \eta_1\|\nabla_{\boldsymbol{h}_1}\mathcal{L}\|_2\|\boldsymbol{x}\|_2^2 = \|\Delta\boldsymbol{W}_1\|\|\boldsymbol{x}\|_2,$$

where the last equality makes use of the spectral norm of rank-1 matrices:

$$\|\Delta\boldsymbol{W}_1\| = \|\eta_1\nabla_{\boldsymbol{h}_1}\mathcal{L}\boldsymbol{x}^\top\| = \eta_1\|\nabla_{\boldsymbol{h}_1}\mathcal{L}\|_2\|\boldsymbol{x}\|_2.$$

Applying this same logic recursively shows that the bounds on $\|\Delta\boldsymbol{h}_2\|_2$ and $\|\Delta\boldsymbol{h}_3\|_2$ are tight as well.

[2] From Lecture 4 we discussed the distribution of eigenvalues of $\frac{1}{M}\boldsymbol{W}^\top\boldsymbol{W}$, which are the same as the singular values of $\frac{1}{\sqrt{M}}\boldsymbol{W}$

2.4 Achieving the Sufficient Spectral Conditions

How do we achieve these spectral conditions? The conditions on $\|\boldsymbol{W}_i\|$ are straightforward. Using the reasoning above about the concentration of spectral norms:[3]

$$\|\boldsymbol{W}_1\| \approx \sigma_1 \left(\sqrt{D} + \sqrt{P}\right), \qquad \|\boldsymbol{W}_2\| \approx \sigma_2 \left(2\sqrt{D}\right), \qquad \|\boldsymbol{W}_3\| \approx \sigma_3 \left(\sqrt{D}\right),$$

we can thus achieve t norms with an appropriate σ_i value:

$$\begin{aligned}
\sigma_1 &= \Theta\left(\sqrt{1/P}\right) \quad \Rightarrow \quad \|\boldsymbol{W}_1\| = \Theta\left(\sqrt{D/P}\right), \\
\sigma_2 &= \Theta\left(\sqrt{1/D}\right) \quad \Rightarrow \quad \|\boldsymbol{W}_2\| = \Theta\left(1\right), \\
\sigma_3 &= \Theta\left(1/D\right) \quad \Rightarrow \quad \|\boldsymbol{W}_3\| = \Theta\left(\sqrt{1/D}\right).
\end{aligned}$$

The condition on $\|\Delta \boldsymbol{W}_i\|$ is a bit more tedious to achieve. We begin with an analysis of the magnitude of the gradients. Using the chain rule/backpropagation, we have:

$$\begin{aligned}
\nabla_{\boldsymbol{W}_3} \mathcal{L} &= \left(\nabla_{h_3} \mathcal{L}\right) \boldsymbol{h}_2 = \left(\nabla_{h_3} \mathcal{L}\right) \boldsymbol{W}_2 \boldsymbol{W}_1 \boldsymbol{x} \\
\nabla_{\boldsymbol{h}_2} \mathcal{L} &= \left(\nabla_{h_3} \mathcal{L}\right) \boldsymbol{W}_3^\top \\
\nabla_{\boldsymbol{W}_2} \mathcal{L} &= \left(\nabla_{\boldsymbol{h}_2} \mathcal{L}\right) \boldsymbol{h}_1^\top = \left(\nabla_{\boldsymbol{h}_3} \mathcal{L}\right) \boldsymbol{W}_3^\top \boldsymbol{h}_1^\top = \left(\nabla_{\boldsymbol{h}_3} \mathcal{L}\right) \boldsymbol{W}_3^\top \boldsymbol{W}_1 \boldsymbol{x} \\
\nabla_{\boldsymbol{h}_1} \mathcal{L} &= \left(\nabla_{\boldsymbol{h}_2} \mathcal{L}\right) \boldsymbol{W}_2^\top = \left(\nabla_{\boldsymbol{h}_3} \mathcal{L}\right) \boldsymbol{W}_3^\top \boldsymbol{W}_2^\top \\
\nabla_{\boldsymbol{W}_2} \mathcal{L} &= \left(\nabla_{\boldsymbol{h}_1} \mathcal{L}\right) \boldsymbol{x}^\top = \left(\nabla_{\boldsymbol{h}_3} \mathcal{L}\right) \boldsymbol{W}_3^\top \boldsymbol{W}_2^\top \boldsymbol{x}.
\end{aligned}$$

Assuming that $\nabla_{h_3} \mathcal{L} = \Theta(1)$ and that we have set $\sigma_1, \sigma_2, \sigma_3$ to achieve the desired spectral norm on $\boldsymbol{W}_1, \boldsymbol{W}_2, \boldsymbol{W}_3$, we can apply the same reasoning as we did when analyzing the bounds of $\boldsymbol{W}_i \boldsymbol{h}_i$ to show that:

$$\begin{aligned}
\|\nabla_{\boldsymbol{W}_3} \mathcal{L}\| &= |\nabla_{h_3} \mathcal{L}| \|\boldsymbol{W}_2\| \|\boldsymbol{W}_1\| \|\boldsymbol{x}\|_2 = \Theta\left(\sqrt{D}\right) \\
\|\nabla_{\boldsymbol{W}_2} \mathcal{L}\| &= |\nabla_{h_3} \mathcal{L}| \|\boldsymbol{W}_3\| \|\boldsymbol{W}_1\| \|\boldsymbol{x}\|_2 = \Theta\left(1\right) \\
\|\nabla_{\boldsymbol{W}_1} \mathcal{L}\| &= |\nabla_{h_3} \mathcal{L}| \|\boldsymbol{W}_3\| \|\boldsymbol{W}_2\| \|\boldsymbol{x}\|_2 = \Theta\left(\sqrt{P/D}\right)
\end{aligned}$$

Thus, recalling that $\Delta \boldsymbol{W}_i = -\eta_i \nabla_{\boldsymbol{W}_i} \mathcal{L}$, we can achieve the desired spectral norms on $\Delta \boldsymbol{W}_i$ by setting the learning rates η_i appropriately:

$$\begin{aligned}
\eta_1 &= \Theta\left(D/P\right) \quad \Rightarrow \quad \|\Delta \boldsymbol{W}_1\| = \Theta\left(\sqrt{D/P}\right), \\
\eta_2 &= \Theta\left(1\right) \quad \Rightarrow \quad \|\Delta \boldsymbol{W}_2\| = \Theta\left(1\right), \\
\eta_3 &= \Theta\left(1/D\right) \quad \Rightarrow \quad \|\Delta \boldsymbol{W}_3\| = \Theta\left(\sqrt{1/D}\right).
\end{aligned}$$

2.5 Zooming Out

Taking everything together, we have demonstrated that one iteration of gradient descent produces meaningful updates to hidden features (i.e. $\Delta[h_i]_j = \Theta(1)$). Contrast this scenario with the linearized regime,

[3] Technically $\boldsymbol{W}_3$ is a vector, so it is not suited for a high-dimensional asymptotic analysis. However, since it is a vector, its spectral norm simply becomes the vector 2-norm which will concentrate around $\sqrt{D}\sigma_3$.

where the hidden features barely changed during training (yet—from the power of averaging—would still yield low loss). We consider the meaningful updates to hidden features as a form of "feature learning."

While we have only sketched informal results in a very simplified training scenario, [Yang and Hu, 2021] derive rigorous feature learning results for any neural network architecture over the entire course of training.

---

3) Open Questions

There are several open questions from our analysis:

1. What do the resulting features converge to?
2. What is the risk associated with the feature learned neural network?

These are challenging questions to answer because we lose a connection with (overparameterized) linear regression. [Woodworth et al., 2020] suggest that the learned features minimize a sparse functional norm, though there is no closed form for this solution. Without a closed form solution it is challenging to characterize the risk, though so-called **mean-field analyses** [e.g. Mei et al., 2018] have begun to characterize the risk in simple models. These analysis, coupled with the overwhelming empirical evidence in favour of feature learning, shows that we still have far to go in understanding why neural networks work.

References

L. Chizat, E. Oyallon, and F. Bach. On lazy training in differentiable programming. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

S. Fort, G. K. Dziugaite, M. Paul, S. Kharaghani, D. M. Roy, and S. Ganguli. Deep learning versus kernel learning: an empirical study of loss landscape geometry and the time evolution of the neural tangent kernel. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

B. Hanin and M. Nica. Finite depth and width corrections to the neural tangent kernel. In *International Conference on Learning Representations*, 2020.

M. Li, M. Nica, and D. Roy. The future is log-Gaussian: ResNets and their infinite-depth-and-width limit at initialization. In *Advances in Neural Information Processing Systems*, volume 34, 2021.

S. Mei, A. Montanari, and P.-M. 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.

P. Savarese, I. Evron, D. Soudry, and N. Srebro. How do infinite width bounded norm networks look in function space? In *Conference on Learning Theory*, pages 2667–2690, 2019.

B. Woodworth, S. Gunasekar, J. D. Lee, E. Moroshko, P. Savarese, I. Golan, D. Soudry, and N. Srebro. Kernel and rich regimes in overparametrized models. In *Conference on Learning Theory*, pages 3635–3673. PMLR, 2020.

G. Yang. Scaling limits of wide neural networks with weight sharing: Gaussian process behavior, gradient independence, and neural tangent kernel derivation. *arXiv preprint arXiv:1902.04760*, 2019.

G. Yang and E. J. Hu. Feature learning in infinite-width neural networks. In *International Conference on Learning Representations*, 2021.

G. Yang, J. B. Simon, and J. Bernstein. A spectral condition for feature learning. *arXiv preprint arXiv:2310.17813*, 2023.

G. Yehudai and O. Shamir. On the power and limitations of random features for understanding neural networks. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

S. Zagoruyko and N. Komodakis. Wide residual networks. In *British Machine Vision Conference*, 2016.