
Implicit Bias in Deep Linear Classification: Initialization Scale vs Training Accuracy

Edward Moroshko
edward.moroshko@gmail.com
Technion

Blake Woodworth
blake@ttic.edu
TTI Chicago

Suriya Gunasekar
suriya@ttic.edu
Microsoft Research

Jason D. Lee
jasonlee@princeton.edu
Princeton University

Nathan Srebro
nati@ttic.edu
TTI Chicago

Daniel Soudry
daniel.soudry@gmail.com
Technion

Abstract

We provide a detailed asymptotic study of gradient flow trajectories and their implicit optimization bias when minimizing the exponential loss over “diagonal linear networks”. This is the simplest model displaying a transition between “kernel” and non-kernel (“rich” or “active”) regimes. We show how the transition is controlled by the relationship between the initialization scale and how accurately we minimize the training loss. Our results indicate that some limit behaviors of gradient descent only kick in at ridiculous training accuracies (well beyond 10^{-100}). Moreover, the implicit bias at reasonable initialization scales and training accuracies is more complex and not captured by these limits.

1 Introduction

The optimization trajectory, and in particular the “implicit bias” determining which predictor the optimization algorithm leads to, plays a crucial role in learning with massively under-determined models, including deep networks, where many zero-error predictors are possible [e.g., 22, 23, 29, 31]. Indeed, in several models we now understand how rich and natural implicit bias, often inducing sparsity of some form, can arise when training a multi-layer network with gradient descent [3, 11, 13, 16, 18–20, 29]. This includes low ℓ_1 norm [30], sparsity in the frequency domain [13], low nuclear norm [11, 18], low rank [3, 25], and low higher-order total variations [6]. A different line of works focuses on how, in a certain regime, the optimization trajectory of neural networks, and hence also the implicit bias, stays near the initialization and mimics that of a kernelized linear predictor (with the kernel given by the tangent kernel) [1, 2, 4, 5, 7–9, 14, 17, 32]. In such a “kernel regime” the implicit bias corresponds to minimizing the norm in some Reproducing Kernel Hilbert Space (RKHS), and cannot yield the rich sparsity-inducing¹ inductive biases discussed above, and is perhaps not as “adaptive” as one might hope. It is therefore important to understand when and how learning is in the kernel regime, what hyper-parameters (e.g., initialization, width, etc.) control the transition out and away from the kernel regime, and how the implicit bias (and thus the inductive bias driving learning) changes in different regimes and as a function of different hyper-parameters.

Initial work identified the **width** as a relevant hyper-parameter, where the kernel regime is reached when the width grows towards infinity [2, 8, 9, 14, 17, 32]. But subsequent work by Chizat et al. [7] pointed to the **initialization scale** as the relevant hyper-parameter, showing that models of any

¹Sparsity in an implicit space can also be understood as feature search or “adaptation”. E.g. eigenvalue sparsity is equivalent to finding features which are linear combinations of input features, and sparsity in the space of ReLU units [e.g., 26] corresponds to finding new non-linear features.

width enter the kernel regime as the scale of initialization tends towards infinity. Follow-up work by Woodworth et al. [30] studied this transition in detail for regression with “diagonal linear networks” (see Section 3), showing how it is controlled by the interactions of **width**, **scale of initialization** and also **depth**. They obtained an exact expression for the implicit bias in terms of these hyper-parameters, showing how it transitions from ℓ_2 (kernel) implicit bias when the width or initialization go to infinity, to ℓ_1 (“rich” or “active” regime) for infinitesimal initialization, showing how a width of k has an equivalent effect to increasing initialization scale by a factor of $\sqrt{k}$. Therefore, studying the effect of initialization scale can also be understood as studying the effect of width.

In this paper, we explore the transition and implicit bias for classification (as opposed to regression) problems, and highlight the effect of another hyper-parameter: **training accuracy**. A core distinction is that in classification with an exponentially decaying loss (such as the cross-entropy/logistic loss), the “rich regime” (e.g., ℓ_1) implicit bias is attained with *any finite initialization*, and not only with infinitesimal initialization as in regression [19, 20], and the scale of initialization does *not* effect the asymptotic implicit bias [12, 15, 28, 29]. This is in seeming contradiction to the fact that even for classification problems, the kernel regime *is* attained when the scale of initialization grows [7]. How can one reconcile the implicit bias not depending on the scale of initialization, with the kernel regime (and thus RKHS implicit bias) being reached as a function of initialization scale? The answer is discussed in detail in Section 2 and depicted in Figure 1: with the decaying losses used in classification, we can never get the loss to zero, only drive it arbitrarily close to zero. If we train infinitely long, then indeed we will eventually reach “rich” bias even for arbitrarily large initialization. On the other hand, if we only consider optimizing to within some arbitrarily high accuracy (e.g., to loss 10^{-10}), we will find ourselves in the kernel regime as the initialization scale grows. But how do these hyper-parameters interact? Where does this transition happen? How accurately do we need to optimize to exit the “kernel” regime? To get the “rich” implicit bias? And what happens in between? What happens if both initialization scale and training accuracy go to infinity? And how is this affected by the depth?

	$\alpha < \infty$	$\alpha \rightarrow \infty$
$\epsilon > 0$		Kernel [7]
$\epsilon \rightarrow 0$	Rich [19]	This Work

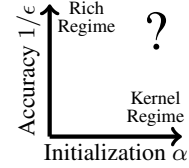


Figure 1: Kernel and rich limits

To answer these questions quantitatively, we consider a concrete model, and study classification using diagonal linear networks. This is arguably the simplest model that displays such transitions,² and as such has already been used to understand non-RKHS implicit bias and dynamics [10, 30].

We consider minimizing the exp-loss of a D -layer diagonal linear network on separable data, starting from initialization at scale α and optimizing until the training loss reaches a value of ϵ . In this case, the two extreme regimes depicted in Figure 1 correspond to implicit biases given by the ℓ_2 norm in the kernel regime and by the $\ell_{2/D}$ quasi-norm when $\epsilon \rightarrow 0$ for fixed α . We consider $\alpha \rightarrow \infty$ and optimizing to within $\epsilon(\alpha) \rightarrow 0$ and ask what happens for different joint behaviours of α and ϵ .

- We identify $\epsilon(\alpha) = \exp(-\Theta(\alpha^D))$ as the boundary of the kernel regime. When $\epsilon(\alpha) = \exp(-o(\alpha^D))$, the optimization trajectory follows that of a kernel machine, and the implicit bias is given by the ℓ_2 norm. But when $\epsilon(\alpha) = \exp(-\Omega(\alpha^D))$, the trajectory deviates from this kernel behaviour.
- Under additional condition (concerning stability of the support vectors), we characterize the behaviour at the transition regime, when $\epsilon(\alpha) = \exp(-\Theta(\alpha^D))$, and show that at this scaling, the implicit bias is given by Woodworth et al.’s Q^D regularizer, which interpolates between the ℓ_1 -norm and the ℓ_2 -norm. This indicates roughly the following optimization trajectory for large enough initialization scale and under the support vector stability condition: we will first pass through the minimum $Q_\infty^D = \ell_2$ -norm predictor (or more accurately, max-margin w.r.t. the ℓ_2 -norm), then traverse the path of minimizers of Q_μ^D , for $\mu \in (\infty, 0)$, until we reach the minimum $Q_0^D = \ell_1$ -norm predictor. We confirm such behaviour in simulations, as well as deviations from it when the condition does not hold.
- As suggested by our asymptotic theory, simulations in Section 5 show that even at moderate initialization scales, extreme training accuracy is needed in order to reach the asymptotic

²Our guiding principal is that when studying a new or not-yet-understood phenomena, we should first carefully and fully understand it in the simplest model that shows it, so as not to get distracted by possible confounders, and to enable a detailed analytic understanding.

behaviour where the implicit bias is well understood. Rather, we see that even in tiny problems, at moderate and reasonable initialization scales and training accuracies, the implicit bias behaves rather different from both the “kernel” and “rich” extremes, suggesting that further work is necessary to understand the effective implicit bias in realistic settings.

Notation For vectors $\mathbf{z}, \mathbf{v}$, we denote by $\mathbf{z}^k$, $\exp(\mathbf{z})$, and $\log(\mathbf{z})$ the element-wise k th power, exponential, and natural logarithm, respectively; $\mathbf{z} \circ \mathbf{v}$ denotes element-wise multiplication; and $\mathbf{z} \propto \mathbf{v}$ implies that $\mathbf{z} = \gamma \mathbf{v}$ for a positive scalar γ . $\mathbf{1}$ denotes the vector of all ones.

2 Kernel and rich regimes in classification

We consider models as mappings $f : \mathbb{R}^p \times \mathcal{X} \rightarrow \mathbb{R}$ from trainable parameters $\mathbf{u} \in \mathbb{R}^p$ and input $\mathbf{x} \in \mathcal{X}$ to predictions. We denote by $F(\mathbf{u}) : \mathbf{x} \mapsto f(\mathbf{u}, \mathbf{x})$ the function implemented by the parameters $\mathbf{u}$. We will focus on models that are D -homogeneous in $\mathbf{u}$ for some $D > 0$, i.e., such that $\forall c > 0, F(c\mathbf{u}) = c^D F(\mathbf{u})$. This includes depth- D linear and ReLU networks.

We consider minimizing $\mathcal{L}(\mathbf{u}) = \frac{1}{N} \sum_{n=1}^N \ell(f(\mathbf{u}, \mathbf{x}_n), y_n)$ for a given dataset $\{(\mathbf{x}_n, y_n) : n = 1, 2, \dots, N\}$ where $\ell : \mathbb{R} \times \mathcal{Y} \rightarrow \mathbb{R}$ is a loss function. We will be mostly focus on binary classification problems, where $y_n \in \{-1, 1\}$, and with the exp-loss $\ell(\hat{y}, y) = \exp(-\hat{y}y)$, which has the same tail behaviour and thus similar asymptotic properties as the logistic or cross-entropy loss [e.g., [19, 28, 29]]. All our results and discussion refer to the exp-loss unless explicitly stated otherwise. We are concerned with understanding the trajectory of gradient descent, which we consider at the limit of infinitesimal stepsize, yielding the gradient flow dynamics,

$$\dot{\mathbf{u}}(t) = -\nabla \mathcal{L}(\mathbf{u}(t)), \quad (1)$$

where here and throughout $\dot{\mathbf{u}} = \frac{d\mathbf{u}}{dt}$.

Along the gradient flow path, $\mathcal{L}(\mathbf{u}(t))$ is monotonically decreasing, and we consider cases where the loss is indeed minimized, i.e., converges to 0 for $t \rightarrow \infty$. However, if we stop the optimization trajectory at a large but finite t , which is what we do in practice, we optimize to some positive training loss $\epsilon(t) = \mathcal{L}(\mathbf{u}(t))$. We define $\tilde{\gamma}(t) = -\log(\epsilon(t))$ as the *training accuracy*. $\tilde{\gamma}(t)$ can also be interpreted as the number of digits of the precision representing the training loss. This is related to the prediction margin $\gamma(t) = \min_n y_n \mathbf{x}_n^\top \mathbf{w}(t)$ as $\gamma(t) \leq \tilde{\gamma}(t) \leq \gamma(t) + \log(N)$ and was introduced as *smoothed margin* in Lyu and Li [19].

For classification problems, we consider **separable data**, i.e., $\exists \mathbf{u}_* \in \mathbb{R}^p : \forall_n y_n f(\mathbf{u}_*, \mathbf{x}_n) > 0$, and so $\mathcal{L}(\mathbf{u}_*) \xrightarrow{\gamma \rightarrow \infty} 0$. But especially in high dimensions, there are many such separating predictors. If $\mathcal{L}(\mathbf{u}(t)) \rightarrow 0$, which of these does $\mathbf{u}(t)$ converge to? Of course $\mathbf{u}(t)$ does not converge, since to approach zero error it must *diverge*. Therefore, instead of the limit of the parameters, we study the limit of the *decision boundary* of the resulting classifier, which is given by $F\left(\frac{\mathbf{u}(t)}{\|\mathbf{u}(t)\|}\right)$.

Kernel Regime When the gradients $\nabla_{\mathbf{u}} f(\mathbf{u}, \mathbf{x}_n)$ do not change much during optimization, then $\mathbf{u}(t)$ behaves as if optimizing over a linearized model $\bar{f}(\mathbf{u}, \mathbf{x}) = f(\mathbf{u}(0), \mathbf{x}) + \langle \mathbf{u} - \mathbf{u}(0), \phi(\mathbf{x}) \rangle$ where $\phi(\mathbf{x}) = \nabla_{\mathbf{u}} f(\mathbf{u}(0), \mathbf{x})$ is the feature map corresponding to the Tangent Kernel at initialization $K(\mathbf{x}, \mathbf{x}') = \langle \phi(\mathbf{x}), \phi(\mathbf{x}') \rangle$ [2, 8, 14]. Consider the trajectory $\bar{\mathbf{u}}(t)$ of gradient flow $\dot{\bar{\mathbf{u}}}(t) = -\nabla \bar{\mathcal{L}}(\bar{\mathbf{u}}(t))$ on the loss of this linearized model $\bar{\mathcal{L}}(\mathbf{u}) = \frac{1}{N} \sum_{n=1}^N \ell(\bar{f}(\mathbf{u}, \mathbf{x}_n), y_n)$. Chizat et al. [7] showed that any D -homogeneous model enters the kernel regime (i.e., behaves like a linearized model) when the scale of the initialization is large:

Theorem 1 (Adapted from Theorem 2.2 in Chizat et al. [7]). *For any fixed time horizon $T > 0$, and any $\mathbf{u}_0$ such that $F(\mathbf{u}_0) = 0$, and for the exp-loss, consider the two gradient flow trajectories $\mathbf{u}(t)$ and $\bar{\mathbf{u}}(t)$, respectively, both initialized with $\mathbf{u}(0) = \bar{\mathbf{u}}(0) = \alpha \mathbf{u}_0$, for $\alpha > 0$. Then $\lim_{\alpha \rightarrow \infty} \sup_{t \in [0, T]} \|F(\mathbf{u}(t)) - \bar{F}(\bar{\mathbf{u}}(t))\| = 0$.*

For a linear model like $\bar{f}$, the gradient flow $\bar{\mathbf{u}}$ converges in direction to the maximum margin solution in the corresponding RKHS norm [28]. Combining this with Theorem 1 we have

$$\lim_{t \rightarrow \infty} \lim_{\alpha \rightarrow \infty} F\left(\frac{\mathbf{u}(t)}{\|\mathbf{u}(t)\|}\right) \propto \operatorname{argmin}_{f: \mathcal{X} \rightarrow \mathbb{R}} \|f\|_K \text{ s.t. } \forall_n : y_n f(\mathbf{x}_n) \geq 1, \quad (2)$$

where recall that K is the Tangent Kernel at the initialization and $\|f\|_K$ is the RKHS norm with respect to this kernel. It is important to highlight the crucial difference here compared to the corresponding statement for the squared loss [7, Theorem 2.3]. For the squared loss we have that $\lim_{\alpha \rightarrow \infty} \sup_{t \in [0, \infty)} \|F(\mathbf{u}(t)) - F(\bar{\mathbf{u}}(t))\| = 0$, i.e., the entire optimization trajectory converges uniformly to $\bar{\mathbf{u}}(t)$. But for the exp-loss, Theorem 1 only ensures convergence for prefixes of the path, up to finite time horizons T . The order of limits in (2) is thus crucial, while for the square loss the order of limits can be reversed.

Rich regime On the other hand, for any finite initialization $\alpha \mathbf{u}_0$, the limit direction of gradient flow, when optimized indefinitely, gives rise to a different limit solution [12, 19, 20]:

Theorem 2 (Paraphrasing Theorem 4.4. in Lyu and Li [19]). *Assume that the gradient flow trajectory in (1) minimizes the loss, i.e., $\mathcal{L}(\mathbf{u}(t)) \rightarrow 0$. Then, any limit point of $\left\{ \frac{\mathbf{u}(t)}{\|\mathbf{u}(t)\|} : t > 0 \right\}$ is along the direction of a KKT point of the following constrained optimization problem:*

$$\min_{\mathbf{u}} \|\mathbf{u}\|_2 \quad \text{s.t. } \forall n : y_n f(\mathbf{u}, \mathbf{x}_n) \geq 1. \quad (3)$$

Compared to the kernel regime in (2), Theorem 2 suggests³ that

$$\lim_{\alpha \rightarrow \infty} \lim_{t \rightarrow \infty} F\left(\frac{\mathbf{u}(t)}{\|\mathbf{u}(t)\|}\right) \propto \text{stationary points of } \min_{f: \mathcal{X} \rightarrow \mathbb{R}} \mathcal{R}(f) \quad \text{s.t. } \forall n : y_n f(\mathbf{x}_n) \geq 1, \quad (4)$$

where $\mathcal{R}(f) = \min_{\mathbf{u}} \|\mathbf{u}\|_2 \quad \text{s.t. } F(\mathbf{u}) = f$.

To understand this double limit, note that Theorem 2 ensures convergence for every α separately, and so also as we take $\alpha \rightarrow \infty$. For neural networks including linear networks, $\mathcal{R}(f)$ captures rich and often sparsity inducing inductive biases (e.g., nuclear norm, higher-order total variations, ℓ_p bridge penalty for $p \leq 1$) that are not captured by RKHS norms [11, 13, 15, 24, 26].

Contrasting (2) and (4) we see that if both the initialization scale α and the optimization time t go to infinity, the order in which we take the limits is crucial in determining the implicit bias, matching the depiction in Figure 1. Roughly speaking, if $\alpha \rightarrow \infty$ first (i.e., faster) and then $t \rightarrow \infty$, we end up in the kernel regime, but if $t \rightarrow \infty$ first (i.e., faster), we can end up with rich implicit bias corresponding to $\mathcal{R}$. The main question we ask is: *where the transition from kernel to rich regime happens, and what is the implicit bias when $\alpha \rightarrow \infty$ and $t \rightarrow \infty$ together?*

Since the time t for gradient flow does not directly correspond to actual “runtime”, and is perhaps less directly meaningful, we instead consider the optimization trajectory in terms of the training loss $\epsilon(t) = \mathcal{L}(\mathbf{u}(t))$. For some loss tolerance parameter ϵ , we follow the gradient flow trajectory until time t such that $\mathcal{L}(t) = \epsilon$ and ask *what is the implicit bias when $\alpha \rightarrow \infty$ and $\epsilon \rightarrow 0$ together?*

3 Diagonal linear network of depth D

In the remainder of the paper we focus on depth- D *diagonal linear networks*. This is a D -homogeneous model with parameters $\mathbf{u} = \begin{bmatrix} \mathbf{u}_+ \\ \mathbf{u}_- \end{bmatrix} \in \mathbb{R}^{2d}$ specified by:

$$f(\mathbf{u}, \mathbf{x}) = \langle \mathbf{u}_+^D - \mathbf{u}_-^D, \mathbf{x} \rangle \quad (5)$$

where recall that the exponentiation is element-wise.

As depicted in Figure 2, the model can be thought of as a depth- D network, with $D - 1$ hidden linear layers (i.e., the output is a weighted sum of the inputs), each consisting of $2d$ units, with the first hidden layer connected to the d inputs and their negations (depicted in the figure as another fixed layer), each unit in subsequent hidden layers connected to only a single unit in the preceding hidden layer, and the single output unit connected to all units in the final hidden layer. That is, the weight

³Theorem 2 is suggestive of $\mathcal{R}(f)$ in (4) as the implicit induced bias in rich regime. However, although global minimizers of (3) and the RHS of (4) are equivalent, the same is not the case for stationary points. For the special cases of certain linear networks and the infinite width univariate ReLU network, stronger results for convergence in direction to the KKT points of (4) can be shown [6, 13, 16].

matrix at each layer $i = 1..D$ is a *diagonal* matrix $\text{diag}(\mathbf{u}^i)$. This presentation has $2dD$ parameters, as every layer has a different weight matrix. However, it is easy to verify that if we initialize all layers to the same weight matrix, *i.e.*, with $\mathbf{u}^i = \mathbf{u}$, then the weight matrices will remain equal to each other throughout training, and so we can just use the $2d$ parameters in $\mathbf{u} \in \mathbb{R}^{2d}$ and take the weight matrix in every layer to be $\text{diag}(\mathbf{u})$, recovering the model (5).

Since all operations in a linear neural net are linear, the model just implements a linear mapping from the input $\mathbf{x}$ to the output, and can therefore be viewed as an alternate parametrization of linear predictors. That is, the functions $F(\mathbf{u}) : \mathcal{X} \rightarrow \mathbb{R}$ implemented by the model is a linear predictor $F(\mathbf{u}) \in \mathcal{X}^*$ (since $\mathcal{X} = \mathbb{R}^d$, we also take $\mathcal{X}^* = \mathbb{R}^d$), and we can write $f(\mathbf{u}, \mathbf{x}) = \langle F(\mathbf{u}), \mathbf{x} \rangle$, where for diagonal linear nets $F(\mathbf{u}) = \mathbf{u}_+^D - \mathbf{u}_-^D$. In particular, for a trajectory $\mathbf{u}(t)$ in parameter space, we can also describe the corresponding trajectory $\mathbf{w}(t) = F(\mathbf{u}(t))$ of linear predictors.

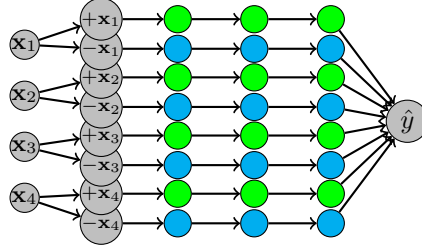


Figure 2: Diagonal linear network of depth 4.

The reason for using both the input features and their negation, and thus $2d$ units per layer, instead of a simpler model with $\mathbf{u} \in \mathbb{R}^d$ and $F(\mathbf{u}) = \mathbf{u}^D$ is two-fold: first, this allows the model to capture mixed sign linear predictors even with even depth. Second, this allows for scaling the parameters while still initializing at the zero predictor. In particular, we will consider initializing to $\mathbf{u}(0) = \alpha \mathbf{1}$, which for the diagonal neural net model (5) corresponds to $\mathbf{w}(0) = F(\mathbf{u}(0)) = 0$ regardless of the scale of α . Such *unbiased initialization* was suggested by Chizat et al. [7] in order to avoid scaling a bias term when scaling the initialization.

Woodworth et al. [30] provided a detailed study of diagonal linear net regression using the square loss $\ell(\hat{y}, y) = (\hat{y} - y)^2$. They showed that for an underdetermined problem (*i.e.*, with multiple zero-error solutions), for any finite α the gradient flow trajectory with squared loss and initialization $\mathbf{u}(0) = \alpha \mathbf{1}$ converges to a zero-error (interpolating) solution minimizing the penalty $Q_{\alpha, D}^D$, where $Q_{\alpha, D}^D$ is:

$$Q_{\mu}^D(\mathbf{w}) = \sum_{i=1}^d q^D \left(\frac{\mathbf{w}_i}{\mu} \right), \text{ where } q^D(z) = \begin{cases} 2 - \sqrt{4 + z^2} + z \cdot \text{arcsinh} \left(\frac{z}{2} \right) & \text{for } D = 2 \\ \int_0^z h_D^{-1}(s) ds & \text{for } D > 2 \end{cases} \quad (6)$$

$$\text{and } h_D(s) = (1 - s)^{-\frac{D}{D-2}} - (1 + s)^{-\frac{D}{D-2}}.$$

For all $D \geq 2$, Q_{μ}^D with $\mu = \alpha^D$ interpolates between the ℓ_1 norm as $\alpha \rightarrow 0$, which corresponds to the rich limit previously shown by Gunasekar et al. [11] and Arora et al. [3], and the ℓ_2 norm as $\alpha \rightarrow \infty$, which is the RKHS norm defined by the tangent kernel at initialization.

Can we identify a similar transition behaviour between kernel and rich regimes with exponential loss?

4 Theoretical Analysis: Between the Kernel and Rich Regimes

We are now ready to state our results that describe the limit behaviour of gradient flow when the initialization scale α and the training accuracy $1/\epsilon$ go to infinity together for classification with linear diagonal networks and the exp-loss on separable data.

First, we establish that if the data is separable, even though the objective $\mathcal{L}(\mathbf{u})$ is non-convex, gradient flow will minimize it, *i.e.*, we will have $\mathcal{L}(\mathbf{u}(t)) \rightarrow 0$. We furthermore obtain a quantitative bound on how fast the training loss decreases as a function of the initialization scale α , the depth D and the ℓ_2 separation margin of the data, $\gamma_2 = \max_{\|\mathbf{w}\|_2=1} \min_n y_n \mathbf{x}_n^\top \mathbf{w}$:

Lemma 3. For $D \geq 2$, any fixed α , and $\forall t$, $\mathcal{L}(\mathbf{u}(t)) \leq \frac{1}{1 + 2D^2 \alpha^{2D-2} \gamma_2^2 t}$.

The proof appears in Appendix D.

We now turn to ask which separating classifier we would get to, if optimizing to within training accuracy ϵ and initialization of scale α , in terms of the relationship between these two quantities. To capture this relationship, we consider a mapping $\epsilon(\alpha)$ such that $\epsilon : \mathbb{R}_{++} \rightarrow (0, 1]$ is strictly monotonic and $\lim_{\alpha \rightarrow \infty} \epsilon(\alpha) = 0$. We call $\epsilon(\alpha)$ the *stopping accuracy function*. For each α , we follow the gradient flow trajectory until time T_α such that $\mathcal{L}(\mathbf{u}(T_\alpha)) = \epsilon(\alpha)$ and denote $\hat{\mathbf{w}}_\alpha = \frac{\mathbf{w}(T_\alpha)}{\gamma(T_\alpha)}$. We study the limit point $\hat{\mathbf{w}} = \lim_{\alpha \rightarrow \infty} \hat{\mathbf{w}}_\alpha$ for different $\epsilon(\alpha)$, assuming this limit exists.

The Kernel Regime We start with showing that if the stopping accuracy function $\epsilon(\alpha)$ goes to zero slowly enough, namely if $\log 1/\epsilon$ is sub-linear in α^D , then with large initialization we obtain the ℓ_2 bias of the kernel regime:

Theorem 4. For $D \geq 2$, if $\epsilon(\alpha) = \exp(-o(\alpha^D))$ then $\hat{\mathbf{w}} = \operatorname{argmin}_{\mathbf{w}} \|\mathbf{w}\|_2$ s.t. $\forall n, y_n \mathbf{w}^\top \mathbf{x}_n \geq 1$.

The proof for $D = 2$ appears in Appendix F.1 and the proof for $D > 2$ appears in Appendix G.1

Escaping the kernel regime Theorem 4 shows that escaping the kernel regime requires optimizing to higher accuracy, such that $\log 1/\epsilon$ is at least linear in the initialization scale α^D . Complementing Theorem 4, we show a converse: that indeed the linear scaling $\log 1/\epsilon = \Theta(\alpha^D)$ is the transition point out of kernel regime, and once $\log 1/\epsilon = \Omega(\alpha^D)$ we no longer obtain the kernel ℓ_2 bias.

To show this, we first identify a condition about the stability of support vectors⁴ for a dataset S .

Condition 5 (Stability condition). A dataset $S = \{(\mathbf{x}_n, y_n) : n = 1, 2, \dots, N\}$ and a stopping accuracy function $\epsilon(\alpha)$ satisfy the stability condition, if for all $k \in [N]$ such that $y_k \mathbf{x}_k^\top \hat{\mathbf{w}} > 1$, and large enough α , there exists $\epsilon^*(\alpha) = \exp(-o(\alpha^D))$ and $\rho_0 > 1$ such that $\forall t$ with $\mathcal{L}(\mathbf{w}(t)) \in [\epsilon(\alpha), \epsilon^*(\alpha)] : \frac{y_k \mathbf{x}_k^\top \mathbf{w}(t)}{\gamma(t)} \geq \rho_0$.

We further say that the stability condition holds uniformly for a given dataset S if there exists ρ_0 such that the above condition holds for all stopping functions $\epsilon(\alpha) = \exp(-\Omega(\alpha^D))$.

If $\mathbf{w}(t)$ indeed follows the trajectory of the linearized model corresponding to the Neural Tangent Kernel, then Condition 5 holds (uniformly) for almost all datasets, see details in Appendix E. Therefore, if Condition 5 does not hold, it follows that we no longer follow the trajectory of this kernel model.

On the other hand, if Condition 5 does hold we show in the following Theorem 6 that when $\epsilon = \exp(-\Theta(\alpha^D))$ we will be in the intermediate regime, leading to max-margin solution with respect to Q^D function in (6), and again deviate from the kernel regime.

Theorem 6. Under Condition 5 for $D \geq 2$, if $\lim_{\alpha \rightarrow \infty} \frac{\alpha^D}{\log(1/\epsilon(\alpha))} = \mu > 0$, then for Q_μ^D as defined in (6), $\hat{\mathbf{w}} = \operatorname{argmin}_{\mathbf{w}} Q_\mu^D(\mathbf{w})$ s.t. $\forall n, y_n \mathbf{x}_n^\top \mathbf{w} \geq 1$.

The proof appears in Appendix F.2 (for $D = 2$) and Appendix G.2 (for $D > 2$). In the proof, we show that the following KKT conditions hold. The result then follows from convexity of $Q_\mu^D(\mathbf{w})$.

$$\exists \nu \in \mathbb{R}_{\geq 0}^N \text{ s.t. } \nabla Q_\mu^D(\hat{\mathbf{w}}) = \sum_{n=1}^N \nu_n y_n \mathbf{x}_n, \quad \forall n : y_n \mathbf{x}_n^\top \hat{\mathbf{w}} \geq 1, \quad \forall n : \nu_n (y_n \mathbf{x}_n^\top \hat{\mathbf{w}} - 1) = 0.$$

The Rich Limit Above we saw that $\log 1/\epsilon$ being linear in α^D is enough to leave the kernel regime, i.e., with this training accuracy and beyond the trajectory no longer behaves as if we were training a kernel machine. We also saw that under Condition 5, when $\log 1/\epsilon$ is exactly linear in α^D , we are in a sense in an “transition” regime, with bias given by the Q_μ^D penalty which interpolates between ℓ_2 (kernel) and ℓ_1 (the rich limit for $D = 2$). Next we show that once the accuracy $\log 1/\epsilon$ is super-linear, and again under Condition 5, we are firmly at the rich limit:

Theorem 7. Under Condition 5 for $D \geq 2$ if $\epsilon(\alpha) = \exp(-\omega(\alpha^D))$ then $\hat{\mathbf{w}} = \operatorname{argmin}_{\mathbf{w}} \|\mathbf{w}\|_1$ s.t. $\forall n, y_n \mathbf{x}_n^\top \mathbf{w} \geq 1$.

For $D = 2$ the result holds also with a weaker condition, when $\epsilon^*(\alpha)$ in Condition 5 is replaced with $\epsilon^*(\alpha) = \exp\left[-o\left(\alpha^2 \log \frac{\log(1/\epsilon(\alpha))}{\alpha^2}\right)\right]$.

The proof for $D = 2$ appears in Appendix F.3 and the proof for $D > 2$ appears in Appendix G.3

For $D > 2$ we know from Theorem 2 that the implicit bias in the rich limit is given by an $\ell_{2/D}$ quasi norm penalty, and not by the ℓ_1 penalty as in Theorem 7. It follows that when Condition 5 holds, the ℓ_1 max-margin predictor must also be a first order stationary point of the $\ell_{2/D}$ max-margin problem. As we demonstrate in Section 5, this is certainly not always the case, and for many problems the ℓ_1

⁴Data-point $(\mathbf{x}_k, y_k)$ is a support vector at time t if $y_k \mathbf{x}_k^\top \mathbf{w}(t) = \min_n y_n \mathbf{x}_n^\top \mathbf{w}(t) = \gamma(t)$.

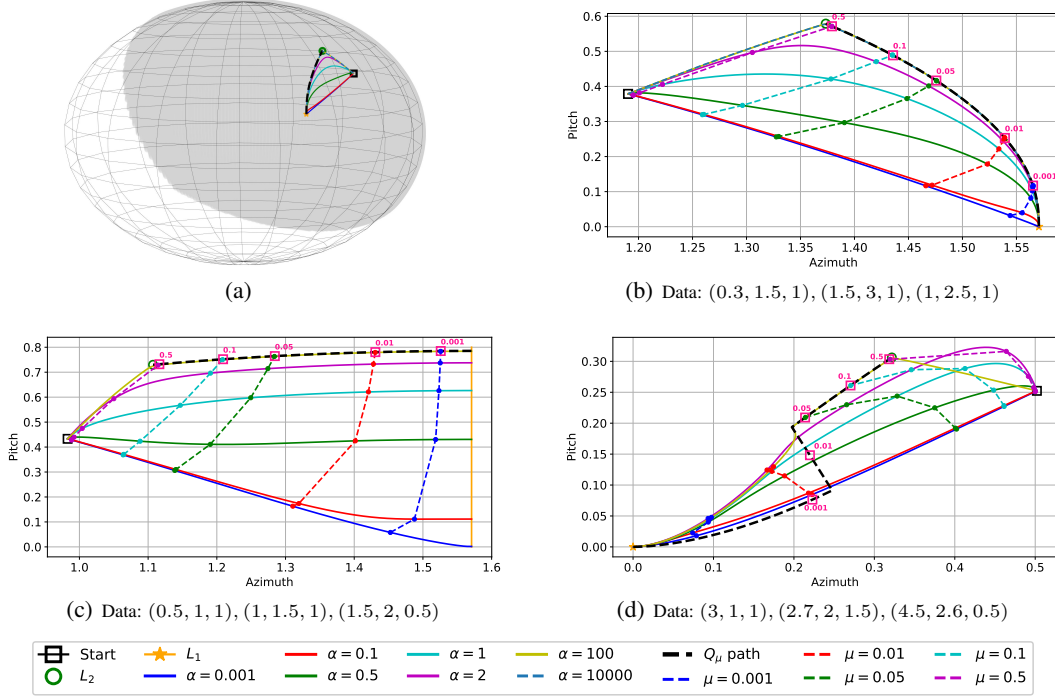


Figure 3: Optimization trajectories for 3 simple datasets in depth 2 linear diagonal network (b-d). Each point in Azimuth-Pitch plane represents a normalized classifier $\mathbf{w}/\|\mathbf{w}\|_2$. The curves corresponding to α in the legend are the entire gradient flow trajectories initialized with the respective α . The curves corresponding to μ are end points of gradient flow trajectories for different α with stopping criteria set as $\epsilon(\alpha) = \exp(-\alpha^2/\mu)$. The pink squares represent the directions along the Q_μ max-margin path for the appropriate μ marked near the square. The dynamics in (b) takes place on a small part of the sphere as shown in (a), where the grey area represents all separating directions.

max-margin is not a stationary point for the $\ell_{2/D}$ max-margin problem—in those cases Condition 5 does not hold. It might well be possible to show that a super-linear scaling is sufficient to reach the rich limit, be it the ℓ_1 max-margin for depth two, or the $\ell_{2/D}$ max-margin (or a stationary point for this non-convex criteria) for higher depth, and we hope future work will address this issue.

Role of depth From Theorem 6 we have that asymptotically $\epsilon(\alpha) = \exp(-\alpha^D/\mu)$. Woodworth et al. [30] analyzed the Q_μ^D function⁵ and concluded that in order to have δ approximation to ℓ_1 limit (achieved for $\mu \rightarrow 0$) we need to have $\mu = \exp(-1/\delta)$ for $D = 2$, and $1/\mu = \text{poly}(1/\delta)$ for $D > 2$. We conclude that in order to have δ approximation to ℓ_1 limit we need the training accuracy to be $\epsilon = \exp(-\alpha^D \exp(1/\delta))$ for $D = 2$, and $\epsilon = \exp(-\alpha^D \text{poly}(1/\delta))$ for $D > 2$. Thus, depth can mitigate the need to train to extreme accuracy. We confirm such behaviour in simulations.

5 Numerical Simulations and Discussion

We numerically study optimization trajectories to see whether we can observe the asymptotic phenomena studied at finite initialization and accuracy. We focus on low dimensional problems, where we can plot the trajectory in the space of predictors. In all our simulations we employ the Normalized GD algorithm, where the gradient is normalized by the loss itself, to accelerate convergence [21]. The learning rate was small enough to ensure gradient flow-like dynamics (always below 10^{-3}).

Gradient flow trajectories In Figure 3 we plot trajectories for training depth $D = 2$ diagonal linear networks in dimension $d = 3$, on several constructed datasets, each consisting of three points.

⁵Note that Q_μ^D is defined for general μ , and $\mu = \alpha^D$ is only for the square loss setting studied in [30]. Here μ depends both on α and ϵ (see Theorem 6).

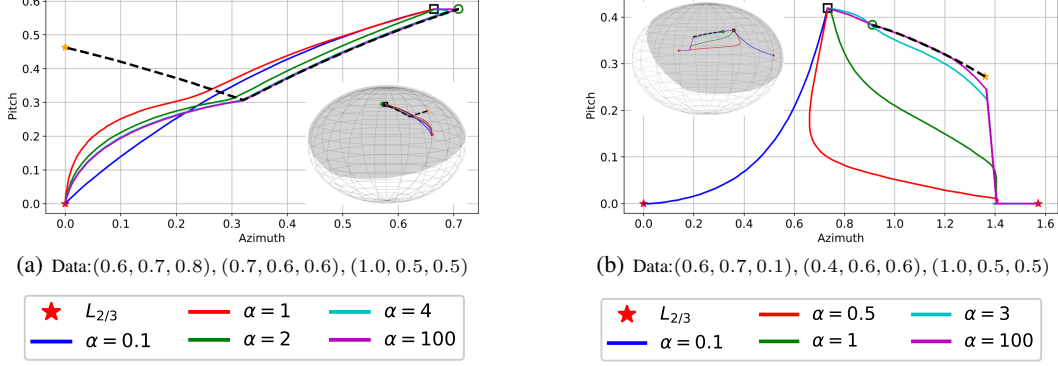


Figure 4: Optimization trajectories for $D = 3$. The legend for ℓ_1 , ℓ_2 , start and Q_μ path is the same as in Figure 3

The trajectory $\mathbf{w}(t)$ in this case is in $\mathbb{R}^3$, and so the corresponding binary predictor given by the normalization $\mathbf{w}(t)/\|\mathbf{w}(t)\|_2$ lies on the sphere (panel (a)). We zoom in on a small section of the sphere and plot the trajectory of $\mathbf{w}(t)/\|\mathbf{w}(t)\|_2$ — the axes correspond to coordinates on the sphere (given as azimuth and pitch). The first step taken by gradient flow will be to the predictor proportional to the average of the data, $\frac{1}{N} \sum_n y_n \mathbf{x}_n$, and we denote this as the “start” point. The grey area in the sphere represents classifiers separating the data (with zero misclassification error), and thus directions where the loss can be driven to zero. The question of “implicit bias” is which of these classifiers the trajectory will converge to. With infinitesimal (small) stepsizes, the trajectories always remain inside this area (*i.e.*, just finding a separating direction is easy), and in a sense, the entire optimization trajectory is driven by the implicit bias.

Panel 3(b) corresponds to a simple situation, with a unique ℓ_1 -max-margin solution, and where the support vectors for the ℓ_2 -max-margin and ℓ_1 -max-margin are the same (although the solutions are not the same!), and so the support vectors do not change throughout optimization and Condition 5 holds uniformly. For large initialization scales ($\alpha = 100$ and $\alpha = 10000$, which are indistinguishable here), the trajectory behaves as the asymptotic theory tells us: from the starting point (average of the data), we first go to the ℓ_2 -max-margin solution (green circle, and recall that this is also the Q_∞^2 -max-margin solution), and then follow the path of Q_μ^2 -max-margin predictors for μ going from ∞ to zero (this path is indicated by the dashed black lines in the plots), finally reaching the ℓ_1 -max-margin predictor (orange star, and this is also the Q_0^2 -max-margin solution). For smaller initialization scales, we still always reach the same endpoint as $\epsilon \rightarrow 0$ (as assured by the theory), but instead of first visiting the ℓ_2 -max-margin solution and traversing the Q_μ^2 path, we head more directly to the ℓ_1 -max-margin predictor. This can be thought of as the effect of initialization on the implicit bias and kernel regime transition: with small initialization we will never see the kernel regime, and go directly to the “rich” limit, but with large initialization we will initially remain in the kernel regime (heading to the ℓ_2 -max-margin), and then, only when the optimization becomes very accurate, escape it gradually.

To see the relative effect of scale and training accuracy, and following our theory, we plot for different values of μ , the different points along trajectories with initialization α such that we fix the stopping criteria as $\epsilon(\alpha) = \exp(-\alpha^2/\mu)$ (dashed cross-lines in the plots). Our theory indicates that for any value of μ , as $\alpha \rightarrow \infty$, the dashes line would converge to the Q_μ^2 -max-margin (a specific point on the Q_μ^2 path), and this is indeed confirmed in Panel 3(b), where the dashed lines converge to pink squares, which correspond to points along the Q_μ^2 path for the appropriate μ values. The clear correspondence between points with the same relationship μ between initialization and accuracy confirms that also at relatively small initialization scales, this parameter is the relevant relationship between them.

From the value μ we can also extract the actual training accuracy. E.g., we see that for a relatively large initialization scale $\alpha = 100$, escaping the kernel regime (getting to the first pink square just removed from the ℓ_2 -max-margin solution, with $\mu = 0.5$), requires optimizing to loss $\epsilon = \exp(-10^4/0.5) \approx 10^{-8700}$, while getting close to the asymptotic limit of the trajectory (the last pink square, with $\mu = 0.001$) requires $\epsilon \approx 10^{-4 \cdot 10^6}$ (that’s four millions digits of precision). Even with reasonable initialization at scale $\alpha = 1$, getting to this limit requires $\epsilon \approx 10^{-434}$.

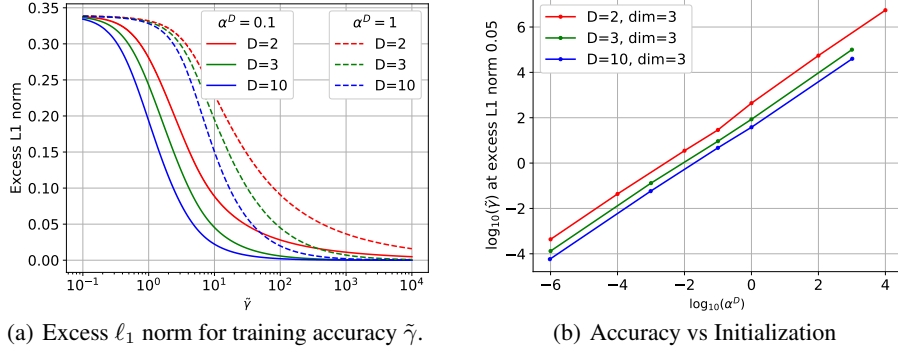


Figure 5: In (a) we plot the excess ℓ_1 norm, defined as $\|\mathbf{w}(t)\|_1 / \|\mathbf{w}_{\ell_1}\|_1 - 1$ where $\mathbf{w}_{\ell_1}$ is the ℓ_1 max-margin (minimum norm) solution, as a function of $\tilde{\gamma}$. For a fixed excess ℓ_1 norm of 0.05, in (b) we plot how long we need to optimize, given some initialization scale, to obtain the 5% closeness to ℓ_1 max-margin solution.

Panel (c) displays the trajectories for another dataset where Condition 5 holds uniformly, but the ℓ_1 -max-margin solution is not unique. Although will always (eventually) converge to a ℓ_1 -max-margin solution, which one we converge to, and thus the implicit bias, *does* depends on the initialization. Panel (d) shows a situation where the support vectors for the ℓ_2 -max-margin and ℓ_1 -max-margin solutions are different, and so change during optimization, and Condition 5 does not hold uniformly. The condition does hold for $\mu > \mu_0$ where $\mu_0 \approx 0.04$, but does not hold otherwise. In this case, even for very large initialization scales α , the trajectory does not follow the Q_μ^2 path entirely. It is interesting to note that the Q_μ^2 is not smooth, which perhaps makes it difficult to follow. Around the kink in the path (at $\mu = \mu_0$), it does seem that larger initialization scales are able to follow it a bit longer, and so perhaps with huge initialization (beyond our simulation ability), the trajectory *does* follow the Q_μ^2 path. Whether the trajectory follows the path for sufficiently large initialization scales even when Condition 5 fails, or whether we can characterize its limit behaviour otherwise, remains an open question.

In Figure 4 we show optimization trajectories for depth 3 linear diagonal network where Condition 5 does not hold uniformly. As we can observe, in both cases the trajectory for large α will first go to ℓ_2 max-margin, then stay near the Q_μ path, until the trajectory starts to deviate in a direction of $\ell_{2/3}$ max-margin solution, whereas the Q_μ path continues to the ℓ_1 max-margin point. In Figure 4(b) we observe that there is a local minimum point at (0,0) and for small α the trajectory converges to it. Further discussion about convergence to local minima in high dimension appears in Appendix H.

Initialization Scale vs Training Accuracy Using the same dataset from Figure 3(b), we examine the question: given some initialization scale, how long we need to optimize to be in the rich regime? Figure 5(a) demonstrates how the initialization and depth affect the convergence rate to the rich regime. Specifically, we chose two reasonable initialization scales α^D , namely 0.1 and 1, and show their convergence to the ℓ_1 max-margin solution as a function of the training accuracy $\tilde{\gamma}$ (recall that $\tilde{\gamma} = \log(1/\epsilon)$), for depths $D = 2, 3, 10$. We chose this dataset so that the rich regime is the same for all depths (*i.e.*, the minimum ℓ_1 norm solution also corresponds to the minimum $\ell_{2/3}$ and $\ell_{2/10}$ quasi-norm solutions).

As previously discussed, even on such a small dataset (with three data-points) we are required to optimize to an incredibly high precision — in order to converge near the rich regime. Notably, the situation is improved and we converge faster to the rich regime when the initialization scale is smaller and/or the depth is larger. Unfortunately, taking the initialization scale to 0 will increase the time needed to escape the vicinity of the saddle point $\mathbf{u} = 0$. For example, Shamir [27] showed that exiting the initialization may take exponential-in-depth time.

In Figure 5(b) we examine the relative scale between (log) accuracy and (log) initialization needed to obtain 5% closeness to ℓ_1 max-margin. Based on the asymptotic result in Theorem 6, we expect that $\tilde{\gamma} \propto \alpha^D$ or $\log(\tilde{\gamma}) = a \log(\alpha^D) + b$, for some constants $a = 1$ and $b \in \mathbb{R}$. And indeed, in Figure 5(b) we obtain $a = 1$, as expected. Note that although our theoretical results are valid for $\alpha \rightarrow \infty$, we obtain the same accuracy vs initialization rate also for small α . Moreover, the intercept of the lines decreases when increasing the depth D , which matches the observed behaviour on Figure 5(a) and to the discussion about the effect of depth in Section 4.

Broader Impact

The goal of this work is to shed light on the implicit bias hidden in the training process of deep networks. These results may enable a better understanding of how hyperparameters select the types of solutions that deep networks converge to, which in turn affect their final generalization performance and hidden biases. This could lead to better performance guarantees or to improved training algorithms which quickly converge to beneficial types of biases. Eventually, we believe progress on these fronts can transform deep learning from the current nascent “alchemy” age (where all the “knobs and levers” of the model and the training algorithm are tuned mostly heuristically during research and development), to a more mature field (like “chemistry”), which can be seamlessly integrated in many real world applications that require high performance, safety, and fair decisions.

Our guiding principal is that when studying a new or not-yet-understood phenomena, we should first study it in the simplest model that shows it, so as not to get distracted by possible confounders, and to enable a detailed analytic understanding, *e.g.*, when understanding or teaching many statistical issues, we would typically start with linear regression, understand the phenomena there, and *then* move on to more complex models. In the specific case here, one of the few models where we have an analytic handle on the implicit bias in the “rich” regime are linear diagonal networks, and it would be very optimistic to hope to get a detailed analytic description of the more complex phenomena we study in models where we can’t even understand the endpoint.

Acknowledgments and Disclosure of Funding

The research of DS was supported by the Israel Science Foundation (grant No. 31/1031), and by the Taub Foundation. This work was partially done while SG, JDL, NS, and DS were visiting the Simons Institute for the Theory of Computing. BW is supported by a Google Research PhD fellowship. JDL acknowledges support of the ARO under MURI Award W911NF-11-1-0303, the Sloan Research Fellowship, and NSF CCF 2002272.

References

- [1] Zeyuan Allen-Zhu, Yuanzhi Li, and Yingyu Liang. Learning and generalization in overparameterized neural networks, going beyond two layers. In *Advances in neural information processing systems*, pages 6155–6166, 2019.
- [2] Zeyuan Allen-Zhu, Yuanzhi Li, and Zhao Song. A convergence theory for deep learning via over-parameterization. In *International Conference on Machine Learning*, pages 242–252, 2019.
- [3] Sanjeev Arora, Nadav Cohen, Wei Hu, and Yuping Luo. Implicit regularization in deep matrix factorization. In *Advances in Neural Information Processing Systems*, pages 7411–7422, 2019.
- [4] 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, 2019.
- [5] Yuan Cao and Quanquan Gu. Generalization bounds of stochastic gradient descent for wide and deep neural networks. In *Advances in Neural Information Processing Systems*, pages 10835–10845, 2019.
- [6] Lénaïc Chizat and Francis Bach. Implicit bias of gradient descent for wide two-layer neural networks trained with the logistic loss. In *Conference on Learning Theory*, pages 1305–1338, 2020.
- [7] Lénaïc Chizat, Edouard Oyallon, and Francis Bach. On lazy training in differentiable programming. In *Advances in Neural Information Processing Systems*, pages 2937–2947, 2019.
- [8] Simon Du, Jason Lee, Haochuan Li, Liwei Wang, and Xiyu Zhai. Gradient descent finds global minima of deep neural networks. In *International Conference on Machine Learning*, pages 1675–1685, 2019.
- [9] Simon S. Du, Xiyu Zhai, Barnabas Póczos, and Aarti Singh. Gradient descent provably optimizes over-parameterized neural networks. In *International Conference on Learning Representations*, 2019.

- [10] Daniel Gissin, Shai Shalev-Shwartz, and Amit Daniely. The implicit bias of depth: How incremental learning drives generalization. In *International Conference on Learning Representations*, 2020.
- [11] Suriya Gunasekar, Blake E Woodworth, Srinadh Bhojanapalli, Behnam Neyshabur, and Nati Srebro. Implicit regularization in matrix factorization. In *Advances in Neural Information Processing Systems*, pages 6151–6159, 2017.
- [12] Suriya Gunasekar, Jason D. Lee, Daniel Soudry, and Nathan Srebro. Characterizing implicit bias in terms of optimization geometry. In *International Conference on Machine Learning*, pages 1827–1836, 2018.
- [13] 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.
- [14] Arthur Jacot, Franck Gabriel, and Clement Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in Neural Information Processing Systems*, pages 8571–8580, 2018.
- [15] Ziwei Ji and Matus Telgarsky. The implicit bias of gradient descent on nonseparable data. In *Conference on Learning Theory*, pages 1772–1798, 2019.
- [16] Ziwei Ji and Matus Jan Telgarsky. Gradient descent aligns the layers of deep linear networks. In *International Conference on Learning Representations*, 2019.
- [17] Yuanzhi Li and Yingyu Liang. Learning overparameterized neural networks via stochastic gradient descent on structured data. In *Advances in Neural Information Processing Systems*, pages 8157–8166, 2018.
- [18] Yuanzhi Li, Tengyu Ma, and Hongyang Zhang. Algorithmic regularization in over-parameterized matrix sensing and neural networks with quadratic activations. In *Conference On Learning Theory*, pages 2–47, 2018.
- [19] Kaifeng Lyu and Jian Li. Gradient descent maximizes the margin of homogeneous neural networks. In *International Conference on Learning Representations*, 2020.
- [20] Mor Shpigel Nacson, Suriya Gunasekar, Jason Lee, Nathan Srebro, and Daniel Soudry. Lexicographic and depth-sensitive margins in homogeneous and non-homogeneous deep models. In *International Conference on Machine Learning*, pages 4683–4692, 2019.
- [21] Mor Shpigel Nacson, Jason D. Lee, Suriya Gunasekar, Pedro Henrique Pamlona Savarese, Nathan Srebro, and Daniel Soudry. Convergence of gradient descent on separable data. In *International Conference on Artificial Intelligence and Statistics*, pages 3420–3428, 2019.
- [22] Behnam Neyshabur. Implicit regularization in deep learning. *arXiv preprint arXiv:1709.01953*, 2017.
- [23] Behnam Neyshabur, Ryota Tomioka, and Nathan Srebro. In search of the real inductive bias: On the role of implicit regularization in deep learning. In *International Conference on Learning Representations, workshop track*, 2015.
- [24] Greg Ongie, Rebecca Willett, Daniel Soudry, and Nathan Srebro. A function space view of bounded norm infinite width relu nets: The multivariate case. In *International Conference on Learning Representations*, 2020.
- [25] Noam Razin and Nadav Cohen. Implicit regularization in deep learning may not be explainable by norms. *arXiv preprint arXiv:2005.06398*, 2020.
- [26] Pedro Savarese, Itay Evron, Daniel Soudry, and Nathan Srebro. How do infinite width bounded norm networks look in function space? In *Conference on Learning Theory*, pages 2667–2690, 2019.
- [27] Ohad Shamir. Exponential convergence time of gradient descent for one-dimensional deep linear neural networks. In *Conference on Learning Theory*, pages 2691–2713, 2019.
- [28] Daniel Soudry, Elad Hoffer, Mor Shpigel Nacson, Suriya Gunasekar, and Nathan Srebro. The implicit bias of gradient descent on separable data. *Journal of Machine Learning Research*, 19(70):1–57, 2018.
- [29] Matus Telgarsky. Margins, shrinkage and boosting. In *International Conference on Machine Learning*, pages 307–315, 2013.

- [30] 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. In *Conference on Learning Theory*, pages 3635–3673, 2020.
- [31] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. In *International Conference on Learning Representations*, 2017.
- [32] Difan Zou, Yuan Cao, Dongruo Zhou, and Quanquan Gu. Stochastic gradient descent optimizes over-parameterized deep relu networks. *Machine Learning Journal*, 2019.