이학석사학위논문

Neural Tangent Kernel Analysis of Deep Narrow Neural Networks

좁고 깊은 심층신경망의 뉴럴 탄젠트 커널 분석

2023년 8월

서울대학교 대학원

수리과학부

이종민

Neural Tangent Kernel Analysis of Deep Narrow Neural Networks

좁고 깊은 심층신경망의 뉴럴 탄젠트 커널 분석

지도교수 Ernest K.Ryu

이 논문을 이학석사 학위논문으로 제출함

2023년 4월

서울대학교 대학원

수리과학부

이 종 민

이종민의 이학석사 학위논문을 인준함

2023년 6월

위 원 장 Otto van Koert (인)

부위원장 Ernest K.Ryu (인)

위 원 박형빈 (인)

An electronic copy of the document has been submitted to the Seoul National University Library with minor formatting adjustments in compliance with the University's requirements.

This dissertation contains contents of the following journal publications:
Name et al., 2021. *J. Good Sci.* 96, 105708;
Name et al., 2021. *Int. J. Nice Sci.* 31, To Appear;
List valid at the time of submission—Material from the dissertation may produce one or more additional publications in the future.

A catalogue record is available from the National Library of Korea

Typeset with LaTeX using cumg-template based on zeta709's snuthesis class
To view the author's profile, visit
`www.yourwebsite.com`

Printed in South Korea
1 3 5 7 8 6 4 2

Abstract

The tremendous recent progress in analyzing the training dynamics of over-parameterized neural networks has primarily focused on wide networks and therefore does not sufficiently address the role of depth in deep learning. In this work, we present the first trainability guarantee of infinitely deep but narrow neural networks. We study the infinite-depth limit of a multilayer perceptron (MLP) with a specific initialization and establish a trainability guarantee using the NTK theory. We then extend the analysis to an infinitely deep convolutional neural network (CNN) and perform brief experiments.

Keywords: Neural tangent kernel, Gradient flow, Deep narrow neural network, Artificial intelligence.

Student ID: 2020-20857

Contents

List of Figures

List of Tables

1 Introduction

Despite the remarkable experimental advancements of deep learning in many domains, a theoretical understanding behind this success remains elusive. Recently, significant progress has been made by analyzing limits of infinitely large neural networks to obtain provable guarantees. The neural tangent kernel (NTK) and the mean-field (MF) theory are the most prominent results.

However, these prior analyses primarily focus on the infinite width limit and therefore do not sufficiently address the role of depth in deep learning. After all, substantial experimental evidence indicates that depth is indeed an essential component to the success of modern deep neural networks. Analyses directly addressing the limit of infinite depth may lead to an understanding of the role of depth.

In this work, we present the first trainability guarantee of infinitely deep but narrow neural networks. We study the infinite-depth limit of a multilayer perceptron (MLP) with a very specific initialization and establish a trainability guarantee using the NTK theory. The MLP uses ReLU activation functions and has width on the order of input dimension + output dimension. Furthermore, we extend the analysis to an infinitely deep convolutional neural network (CNN) and perform brief experiments.

1.1 Prior works

The classical universal approximation theorem establishes that wide 2-layer neural networks can approximate any continuous function [12, 17]. Extensions and generalizations [25, 35, 30, 5, 54] and random feature learning [55, 56, 57], a constructive version of the universal approximation theorem, use large width in their analyses. As overparameterization got recognized as a key component in understanding the performance of deep learning [74], analyses of large neural networks started to appear in the literature [66, 2, 15, 16, 75, 36], and their infinite-width limits such as neural network as Gaussian process (NNGP) [44, 43, 70, 32, 41, 47], neural tangent kernel (NTK) [27], and mean-field (MF) [42, 9, 61, 60, 64, 63, 45, 53] were formulated. NNGP characterizes the neural network at initialization, while NTK and MF analyses provide guarantees of trainability with SGD. This line of research naturally raises the question of whether very deep neural networks also enjoy similar properties as wide neural networks, especially given the importance of depth in modern deep learning.

The analogous line of research for very deep neural networks has a shorter history. Universality of deep narrow neural networks [40, 22, 37, 19, 31, 49, 68], lower bounds on the minimum width necessary for universality [40, 22, 29, 49], and quantitative analyses showing the benefit of depth over width in approximating certain functions [69] are all very recent developments. On the other hand, the neural ODE [7] is a continuous-depth model that can be considered an infinite-depth limit of a neural network with residual connections. Also stochastic extensions of neural ODE, viewing infinitely deep ResNets as diffusion processes, have been considered in [51, 50]. However, these continuous-depth limits do not come with any trainability or generalization guarantees.

In the infinite width *and* depth limit, a quantitative universal approximation result has been established [38] and a trainability guarantee was obtained in setups combining the MF limit with the continuous-depth limit inspired by the neural ODE [39, 13, 14].

Efforts to understand the trainability of deep non-wide neural networks have been made. [3, 62] establishes trainability guarantees for linear deep networks (no activation functions). [52] studied the so-called dynamic isometry property, [20] studied the exploding and vanishing gradient problem for ReLU MLPs, and [26] studied the NTK of deep MLPs and ResNets at initialization, but these results are limited to the state of the neural network at initialization and therefore do not directly establish guarantees on the training dynamics. To the best of our knowledge, no prior work has yet established a trainability guarantee on (non-linear) deep narrow neural networks.

Many extensions and variations of the NTK have been studied: NTK analysis with convolutional layers [4, 73], further refined analyses and experiments [34], NTK analysis with regularizers and noisy gradients [8], finite-width NTK analysis [21], quantitative universality result of the NTK [28], generalization properties of overparameterized neural networks [6, 71], unified analysis of NTK and MF [18], notion of lazy training generalizing linearization effect of the NTK regime [10], closed-form evaluations of NTK kernel values [11, 4], analyses of distribution shift and meta learning in the infinite-width regime [1, 46], library implementations of infinitely wide neural networks as kernel methods with NTK [48], empirical evaluation of finite vs. infinite neural networks [33], and constructing better-performing kernels with improved extrapolation of the standard initialization [65]. Such results have mostly focused on the analysis and application of wide, rather than deep, neural networks.

2 Preliminaries and Notations

In this section, we review the necessary background and set up the notation. We largely follow the notions of [27], although our notation has some minor differences.

Given a function $f_\theta(x) \in \mathbb{R}^n$ with $\theta \in \mathbb{R}^p$ and $x \in \mathbb{R}^d_+$, write $\partial_\theta f_\theta(x) \in \mathbb{R}^{n \times p}$ to denote the Jacobian matrix with respect to θ. If $f_\theta(x) \in \mathbb{R}$ is scalar-valued, write $\nabla_\theta f_\theta(x) \in \mathbb{R}^{p \times 1}$ for the gradient. The gradient and Jacobian matrices are, by convention, transposes of each other, i.e., $\nabla_\theta f_\theta(x) = (\partial_\theta f_\theta(x))^\intercal$. Write $\|\cdot\|$ to denote the standard Euclidean norm for vectors and the standard operator norm for matrices. Write $\langle \cdot, \cdot \rangle$ for the vector and Frobenius inner products. Write $\mathbb{R}^d_+ \subset \mathbb{R}^d$ for the strict positive orthant, i.e., $\mathbb{R}^d_+$ is the set of vectors with element-wise positive entries. Write $\xrightarrow{p}$ to denote convergence in probability. Write $g \sim \mathcal{GP}(\mu, \Sigma)$ to denote that g is a Gaussian process with mean $\mu(x)$ and covariance kernel $\Sigma(x, x')$ [58, 59].

Let p^{in} be the empirical distribution on a training dataset $x_1, x_2, \ldots, x_N \in \mathbb{R}^{d_{in}}_+$, which we assume are element-wise positive. Let $\mathcal{F} = \{f \colon \mathbb{R}^{d_{in}}_+ \to \mathbb{R}^{d_{out}}\}$ be the function space with a seminorm $\|\cdot\|_{p^{in}}$ induced by the bilinear map

$$\langle f, g \rangle_{p^{in}} = \mathbb{E}_{x \sim p^{in}}[f(x)^\intercal g(x)].$$

If W is a matrix-valued function, write

$$\|W\|_{p^{in}}^2 = \mathbb{E}_{x\sim p^{in}} \|W(x)\|^2,$$

where $\|\cdot\|$ here is the operator norm.

2.1 Kernel gradient flow

Let $\mathcal{L}\colon \mathcal{F} \to \mathbb{R}$ be a functional loss. We primarily consider $\mathcal{L}(f) = \frac{1}{2}\|f - f^\star\|_{p^{in}}^2$ with a given target function $f^\star$. We train a neural network f_θ by solving

$$\underset{\theta}{\text{minimize}} \quad \mathcal{L}(f_\theta).$$

Let $\mathcal{F}^*$ denote the dual of $\mathcal{F}$ with respect to p^{in}. So $\mathcal{F}^*$ consists of linear maps $\langle \delta, \cdot \rangle_{p^{in}}$ for some $\delta \in \mathcal{F}$. Let $\partial_f \mathcal{L}|_{f_0}$ denote the functional derivative of the loss at f_0. Since $\partial_f \mathcal{L}|_{f_0} \in \mathcal{F}^*$, there exists a corresponding dual element $\delta|_{f_0} \in \mathcal{F}$, where $\partial_f \mathcal{L}|_{f_0} = \langle \delta|_{f_0}, \cdot \rangle_{p^{in}}$. To clarify, $\delta|_{f_0}(x)$ is defined to be a length d_{out} column vector.

A multi-dimensional kernel $K\colon \mathbb{R}_+^{d_{\text{in}}} \times \mathbb{R}_+^{d_{\text{in}}} \to \mathbb{R}^{d_{\text{out}} \times d_{\text{out}}}$ is a function such that $K(x, x') = K(x', x)^\intercal$ for all $x, x' \in \mathbb{R}_+^{d_{\text{in}}}$. A multi-dimensional kernel is positive semidefinite if

$$\mathbb{E}_{x,x'\sim p^{in}} \left[f(x)^\intercal K(x, x') f(x') \right] \geq 0$$

for all $f \in \mathcal{F}$ and (strictly) positive definite if the inequality holds strictly when $\|f\|_{p^{in}} \neq 0$. To clarify, $\mathbb{E}_{x,x'\sim p^{in}}$ denotes the expectation with respect to x and x' sampled independently from p^{in}. The kernel gradient of $\mathcal{L}$ at f_0 with respect to the kernel K is defined as

$$\nabla_K \mathcal{L}|_{f_0}(x) = \mathbb{E}_{x'\sim p^{in}} \left[K(x, x') \delta|_{f_0}(x') \right]$$

for all $x \in \mathbb{R}_+^{d_{\text{in}}}$. We say a time-dependent function f_t follows the kernel gradient flow with respect to K if

$$\partial_t f_t(x) = -\nabla_K \mathcal{L}|_{f_t}(x)$$

for all $t > 0$ and $x \in \mathbb{R}_+^{d_{\text{in}}}$. During the kernel gradient flow, the loss $\mathcal{L}(f_t)$ evolves as

$$\partial_t \mathcal{L}|_{f_t} = -\mathbb{E}_{x,x' \sim p^{in}} \left[\delta|_{f_t}(x)^\intercal K(x,x') \delta|_{f_t}(x') \right].$$

If K is positive definite and if certain regularity conditions hold, then kernel gradient flow converges a critical point and it converges to a global minimum if $\mathcal{L}$ is convex and bounded from below.

2.2 Neural tangent kernel

Given a neural network $f_{\theta(t)}$, [27] defines the neural tangent kernel (NTK) at time t as

$$\Theta_t(x,x') = \partial_\theta f_{\theta(t)}(x) \left(\partial_\theta f_{\theta(t)}(x') \right)^\intercal,$$

which, by definition, is a positive semidefinite kernel. [27] pointed out that a neural network trained with gradient flow, which we define and discuss in Section 3.1.1, can be viewed as kernel gradient flow with respect to Θ_t, i.e.,

$$\partial_t f_{\theta(t)}(x) = -\nabla_{\Theta_t} \mathcal{L}|_{f_t}(x).$$

However, even though Θ_t is always positive semidefinite, the time-dependence of Θ_t makes the training dynamics non-convex and prevents one from establishing trainability guarantees in general. The contribution of [27] is showing that $\Theta_t \xrightarrow{p} \Theta$ in an appropriate infinite-width limit, where Θ is a fixed limit that

does not depend on time. Then, since Θ is fixed, kernel gradient flow generically converges provided that the loss $\mathcal{L}$ is convex.

3 NTK analysis of infinitely deep MLP

Consider an L-layer multilayer perceptron (MLP) $f_\theta^L \colon \mathbb{R}_+^{d_{\text{in}}} \to \mathbb{R}^{d_{\text{out}}}$ parameterized by θ, where the input $x \in \mathbb{R}_+^{d_{\text{in}}}$ is a d_{in}-dimensional vector with positive entries and the output is a d_{out}-dimensional vector. The network consists of $L-1$ fully connected hidden layers with uniform width $d_{\text{in}} + d_{\text{out}} + 1$, each followed by the ReLU activation function. The final output layer has width d_{out} and is not followed by an activation function.

Let us set up specific notation. Define the pre-activation values as

$$f_{\theta^{(1)}}^1(x) = W^1 x + b^1$$

$$f_{\theta^{(l)}}^l(x) = W^l \sigma(f_{\theta^{(l-1)}}^{l-1}(x)) + b^l, \qquad 2 \le l \le L.$$

We use ReLU for the activation σ. The weight matrices have dimension $W^1 \in \mathbb{R}^{(d_{\text{in}}+d_{\text{out}}+1)\times d_{\text{in}}}$, $W^2, \ldots, W^{L-1} \in \mathbb{R}^{(d_{\text{in}}+d_{\text{out}}+1)\times(d_{\text{in}}+d_{\text{out}}+1)}$, and $W^L \in \mathbb{R}^{d_{\text{out}}\times(d_{\text{in}}+d_{\text{out}}+1)}$. The bias vectors have dimension $b^1, \ldots, b^{L-1} \in \mathbb{R}^{(d_{\text{in}}+d_{\text{out}}+1)\times 1}$ and $b^L \in \mathbb{R}^{d_{\text{out}}\times 1}$. For $1 \le l \le L$, write $\theta^{(l)} = \{W^i, b^i : i \le l\}$ to denote the collection of parameters up to l-th layer. Let $f_\theta^L = f_{\theta^{(L)}}^L$.

3.1 Initialization

Motivated by [31], initialize the weights of our L-layer MLP f_θ^L as follows:

$$W^1 = \begin{bmatrix} C_L I_{d_{\text{in}}} \\ u^1 \\ 0_{d_{\text{out}} \times d_{\text{in}}} \end{bmatrix}$$

$$W^l = \begin{bmatrix} I_{d_{\text{in}}} & 0_{d_{\text{in}} \times 1} & 0_{d_{\text{in}} \times d_{\text{out}}} \\ u^l & 0_{1 \times 1} & 0_{1 \times d_{\text{out}}} \\ 0_{d_{\text{out}} \times d_{\text{in}}} & 0_{d_{\text{out}} \times 1} & I_{d_{\text{out}}} \end{bmatrix}$$

$$W^L = \begin{bmatrix} 0_{d_{\text{out}} \times d_{\text{in}}} & 0_{d_{\text{out}} \times 1} & I_{d_{\text{out}}} \end{bmatrix}$$

for $2 \leq l \leq L-1$, where $u^l \in \mathbb{R}^{1 \times d_{\text{in}}}$ is randomly sampled $u_i^l \overset{iid}{\sim} \mathcal{N}(0, \frac{1}{d_{\text{in}}}\rho^2)$ for $1 \leq l \leq L-1$. Here, I_d is the $d \times d$ identity matrix and $0_{m \times n}$ is the m by n matrix with all zero entries, $C_L > 0$ is a scalar growing as a function of L at a rate satisfying $L^2/C_L \to 0$, and $\rho > 0$ is a fixed variance parameter. Initializes the biases as follows:

$$b^1 = \begin{bmatrix} 0_{d_{\text{in}} \times 1} \\ v^1 \\ C_L 1_{d_{\text{out}}} \end{bmatrix}$$

$$b^l = \begin{bmatrix} 0_{d_{\text{in}} \times 1} \\ v^l \\ 0_{d_{\text{out}} \times 1} \end{bmatrix}$$

$$b^L = \begin{bmatrix} -C_L 1_{d_{\text{out}}} \end{bmatrix}$$

for $2 \leq l \leq L-1$, where $v^l \in \mathbb{R}$ is randomly sampled $v^l \overset{iid}{\sim} \mathcal{N}(0, C_L^2 \beta^2)$ for $1 \leq l \leq L-1$. Here, $\beta > 0$ is a fixed variance parameter. Note that $1_k \in \mathbb{R}^{k \times 1}$

is the vector whose entries are all 1. Figure 3.1 illustrates this initialization scheme.

We clarify that while we use many specific non-random initializations, those parameters are not fixed throughout training. In other words, all parameters are trainable, just as one would expect from a standard MLP.

[31] used a similar construction to establish a universal approximation result for deep MLPs by showing that their deep MLP mimics a 2-layer wide MLP. However, their main concern is the existence of a weight configuration that approximates a given function, which does not guarantee that such a configuration can be found through training. On the other hand, we propose an explicit initialization and establish a trainability guarantee; our network outputs 0 at initialization and converges to the desired configuration through training.

3.1.1 Gradient flow and neural tangent kernel

We are now ready to describe the training of our neural network f_θ^L via gradient flow, a continuous-time model of gradient descent. Since our initialization scales the input by C_L at the first layer, we scale the learning rate accordingly, both in the continuous-time analysis of Section 4 and in the discrete-time experiments of Section 5, so that we get meaningful limits as $L \to \infty$.

Train f_θ^L with

$$\begin{aligned}
\partial_t \theta(t) &\overset{\text{(a)}}{=} -\frac{1}{LC_L^2} \nabla_\theta \text{Loss}(\theta)\Big|_{\theta=\theta(t)} \\
&\overset{\text{(b)}}{=} -\frac{1}{LC_L^2} \left(\partial_\theta \mathcal{L}(f_\theta^L)\right)^\intercal \Big|_{\theta=\theta(t)} \\
&\overset{\text{(c)}}{=} -\frac{1}{LC_L^2} \mathbb{E}_{x\sim p^{in}} \left[(\partial_\theta f_\theta^L(x))^\intercal \delta|_{f_\theta^L}(x)\right] \Big|_{\theta=\theta(t)},
\end{aligned}$$

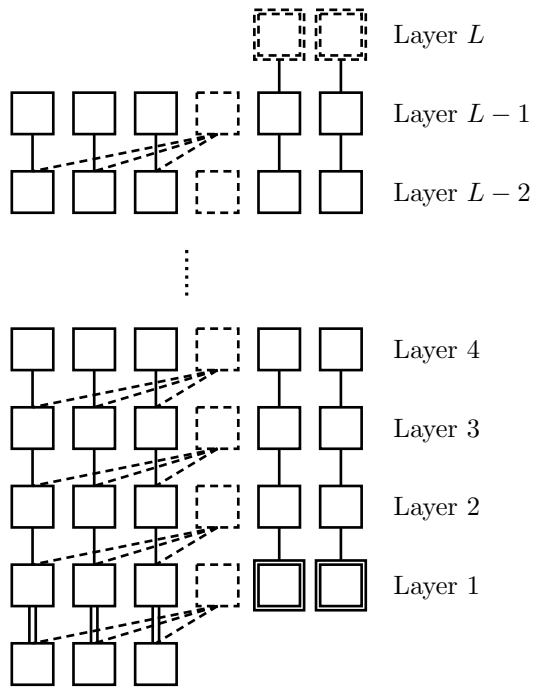


Figure 3.1 Initialization of deep MLP with $d_{in} = 3$ and $d_{out} = 2$. Intermediate layers have width $d_{in} + 1 + d_{out}$. Line styles indicate types of weight initializations (solid:1, double:C_L, dash:Gaussian, none:0). Box styles indicate types of bias initializations (solid:0, dash:Gaussian, double:C_L, double-dash:$-C_L$).

where (a) defines the θ-update to be gradient flow with learning rate $1/(LC_L^2)$, (b) plugs in our notation, and (c) follows from the chain rule.

This gradient flow defines $\theta(t)$ to be a function of time, but we will often write θ rather than $\theta(t)$ for notational conciseness. This gradient flow and the chain rule induces the functional dynamics of the neural network:

$$\begin{aligned}\partial_t f_\theta^L(x) &= \partial_\theta f_\theta^L(x)\partial_t\theta \\ &= -\frac{1}{LC_L^2}\mathbb{E}_{x'\sim p^{in}}\left[\partial_\theta f_\theta^L(x)\left(\partial_\theta f_\theta^L(x')\right)^\intercal \delta|_{f_\theta^L}(x')\right].\end{aligned}$$

Since we use a scaling factor in our gradient flow, we define the *scaled NTK* at time t as

$$\tilde{\Theta}_t^L(x,x') = \frac{1}{LC_L^2}\partial_\theta f_\theta^L(x)\left(\partial_\theta f_\theta^L(x')\right)^\intercal.$$

Then,

$$\begin{aligned}\partial_t f_\theta^L(x) &= -\mathbb{E}_{x'\sim p^{in}}\left[\tilde{\Theta}_t^L(x,x')\delta|_{f_\theta^L}(x')\right] \\ &= -\nabla_{\tilde{\Theta}_t^L}\mathcal{L}|_{f_\theta^L}(x).\end{aligned}$$

3.2 Convergence in infinite-depth limit

We now analyze the convergence of the infinitely deep MLP.

Theorem 3.1 establishes that the scaled NTK at initialization (before training) of the randomly initialized MLP converges to a deterministic limit with a closed-form expression as the depth L becomes infinite.

Theorem 3.1 (Scaled NTK at initialization) *Suppose f_θ^L is initialized as in Section 3.1. For any $x, x' \in \mathbb{R}_+^{d_{in}}$,*

$$\tilde{\Theta}_0^L(x,x') \xrightarrow{p} \tilde{\Theta}^\infty(x,x')$$

as $L \to \infty$*, where*

$$\tilde{\Theta}^{\infty}(x, x') = \left(x^{\intercal}x' + 1 + \mathbb{E}_g[\sigma(g(x))\sigma(g(x'))]\right) I_{d_{\text{out}}},$$

and $g \sim \mathcal{GP}(0, \frac{\rho^2}{d_{\text{in}}}x^{\intercal}x' + \beta^2)$.

When $L < \infty$, the scaled NTK $\tilde{\Theta}_t^L(x, x')$ depends on time through its dependence on $\theta(t)$. Theorem 3.2 establishes that $\tilde{\Theta}_t^L(x, x')$ becomes independent of t as $L \to \infty$.

Theorem 3.2 (Invariance of scaled NTK) *Let* $T > 0$*. Suppose* $\int_0^T \left\| \delta|_{f_\theta^L} \right\|_{p^{in}} dt$ *is stochastically bounded as* $L \to \infty$*. Then, for any* $x, x' \in \mathbb{R}_+^{d_{in}}$*,*

$$\tilde{\Theta}_t^L(x, x') \xrightarrow{p} \tilde{\Theta}^{\infty}(x, x')$$

uniformly for $t \in [0, T]$ *as* $L \to \infty$*.*

Let $\mathcal{L}(f) = \frac{1}{2}\left\|f - f^{\star}\right\|_{p^{in}}^2$ be the quadratic loss. In this case, the stochastic boundedness assumption of Theorem 3.2 holds, as we show in the appendix, and we characterize the training dynamics explicitly. For the sake of notational simplicity, assume the MLP's prediction is a scalar, i.e., assume $d_{out} = 1$. The generalization to multi-dimensional outputs is straightforward, following the arguments of [27, Section 5].

Theorem 3.3 concludes the analysis by characterizing the trained MLP as $L \to \infty$. Define the kernel regression predictor as

$$f_{\text{ntk}}(x) = \left(\tilde{\Theta}^{\infty}(x, x_1), \ldots, \tilde{\Theta}^{\infty}(x, x_N)\right) K^{-1} f^{\star}(X),$$

where $K_{i,j} = \tilde{\Theta}^{\infty}(x_i, x_j)$ and $[f^{\star}(X)]_i = f^{\star}(x_i)$ for $i, j \in \{1, \ldots, N\}$. Let f_t be trained with the limiting kernel gradient flow of $f_{\theta(t)}^L$ as $L \to \infty$, i.e., $f_0 = 0$

and f_t follows

$$\partial_t f_t = -\nabla_{\tilde{\Theta}^\infty} \mathcal{L}|_{f_t}(x).$$

Then the infinitely deep training dynamics converge to f_{ntk} in the following sense.

Theorem 3.3 (Equivalence between deep MLP and kernel regression) *Let* $\mathcal{L}(f) = \frac{1}{2}\|f - f^\star\|_{p^{in}}^2$. *Let* $\tilde{\Theta}^\infty$ *be positive definite. If* f_t *follows the kernel gradient flow with respect to* $\tilde{\Theta}^\infty$, *then for any* $x \in \mathbb{R}_+^{d_{in}}$,

$$\lim_{t\to\infty} f_t(x) = f_{\text{ntk}}(x).$$

3.3 Proof outline

At a high level, our analysis follows the same line of argument as that of the original NTK paper [27]: Theorem 3.1 characterizes the limiting NTK at initialization, Theorem 3.2 establishes that the NTK remains invariant throughout training, and Theorem 3.3 establishes convergence in the case of quadratic loss functions. The proof of Theorem 3.3 follows from arguments similar to those of [27, Theorem 3]. The proof of Theorem 3.1 follows from identifying the recursive structure and noticing that the initialization is designed to simplify this recursion.

The key technical challenge of this work is in Theorem 3.2. The analysis is based on defining the Lyapunov function

$$\Gamma_L(t) = \Psi_{L,2}(t) + \Psi_{L,8}(t) + \sum_{j=1}^{8} \Phi_{L,j}(t)$$

(the individual terms will be defined soon), establishing

$$\Gamma_L(t) \le \Gamma_L(0) + \int_0^t \mathcal{O}(L/C_L)\Gamma_L(s)^4\, ds,$$

and appealing to Grönwall's lemma to show that $\Gamma_L(t)$ is invariant, i.e., $\Gamma_L(t) \to \Gamma_\infty(0)$ as $L \to \infty$ uniformly in $t \in [0,T]$. For $j = 1, \ldots, 8$, the first term is defined as

$$\Phi_{L,j} = \frac{1}{(L-1)C_L^j} \sum_{l=1}^{L-1} \left(\left\| f^l_{\theta^{(l)}(0)} \right\|_{p^{in}} + \left\| f^l_{\theta^{(l)}(t)} - f^l_{\theta^{(l)}(0)} \right\|_{p^{in}} \right)^j$$

and its invariance implies that the average variation of the layers vanish for inputs $x \in \{x_1, \ldots, x_N\}$ to the MLP.

Because we study the infinite-depth regime, our Lyapunov analysis is significantly more technical compared to the prior NTK analyses studying the infinite-width regime. Prior work dealt with sums of infinitely many terms, which were analyzed with the central limit theorem and law of large numbers. In contrast, the infinite depth of our setup leads to an infinite composition of layers and an infinite product of matrices, which must be controlled through a more delicate Lyapunov analysis.

For $L \geq j \geq k \geq 2$, define

$$\mathfrak{W}^k_j(x,t) = W^j \mathrm{diag}(\dot\sigma(f^{j-1}_{\theta^{(j-1)}})) \cdots W^k \mathrm{diag}(\dot\sigma(f^{k-1}_{\theta^{(k-1)}}))$$

(where W^j and $\theta^{(j)}$ depend on t and $f^l_{\theta^{(l)}}$ depends on x and t) and we bound its change by incorporating the following terms into the Lyapunov function:

$$\Psi_{L,2} = \tfrac{1}{(L-1)^2} \textstyle\sum_{l=2}^{L} \frac{L-1}{l-1} \sum_{i=2}^{l} \left(\left\| \mathfrak{W}^i_l(\cdot,0) \right\|_{p^{in}} + \left\| \mathfrak{W}^i_l(\cdot,t) - \mathfrak{W}^i_l(\cdot,0) \right\|_{p^{in}} \right)^2$$

$$\Psi_{L,8} = \tfrac{1}{L-1} \textstyle\sum_{l=2}^{L} \left(\left\| \mathfrak{W}^l_L(\cdot,0) \right\|_{p^{in}} + \left\| \mathfrak{W}^l_L(\cdot,t) - \mathfrak{W}^l_L(\cdot,0) \right\|_{p^{in}} \right)^8.$$

Establishing the invariance of $\Gamma_L(t)$ as $L \to \infty$ is the first step of the analysis, but, by itself, it does not characterize the limiting MLP for inputs $x \notin \{x_1, \ldots, x_N\}$ and it only bounds the average variation of the layers, rather than the layer-wise variation. Hence, we generalize the invariance results with

two additional Lyapunov analyses and combine these results to establish the scaled NTK's invariance.

Our Lyapunov analysis significantly is simplified by setting $\ddot{\sigma}(s) = 0$ and thereby removing terms involving $\ddot{\sigma}$. However, while $\ddot{\sigma}(s) = 0$ for $s \neq 0$, we cannot ignore the fact that $\ddot{\sigma}(0) \neq 0$ (in fact undefined). We resolve this issue by showing that all instances of $\sigma(s)$ never encounter the input $s = 0$ throughout training with probability approaching 1 as $L \to \infty$. We outline this argument below.

Due to the randomness of the initialization, the pre-activation values $f^l_{\theta^{(l)}(0)}(x)$ are element-wise nonzero for all $x \in \{x_1, \ldots, x_N\}$ with probability 1. If $(f^l_{\theta^{(l)}(t)}(x))_r = 0$ for some l, r, and t, i.e., if a *zero-crossing* happens for a neuron at time t, then the analysis must somehow deal with the behavior of $\sigma(s)$ at $s = 0$. However, if zero-crossing happens for no neurons for all $t \in [0, T]$, then we can safely set $\ddot{\sigma} = 0$ in our analysis. We prove that the probability of a zero crossing (over all neurons of all layers and all $t \in [0, T]$) vanishes as $L \to \infty$. We specifically establish this claim by showing that as $L \to \infty$,

$$\sup_{1 \leq l \leq L-1, t \in [0,T]} \left\| f^l_{\theta^{(l)}(t)}(x) - f^l_{\theta^{(l)}(0)}(x) \right\| \leq KL$$

with high probability and

$$\Pr\left[\inf_{l,r} \left| \left(f^l_{\theta^{(l)}(0)}(x) \right)_r \right| > (K+1)L \right] \to 1.$$

for some constant $K > 0$. These two results establish that the zero-crossing probability vanishes as $L \to \infty$. We provide the details in Appendix.

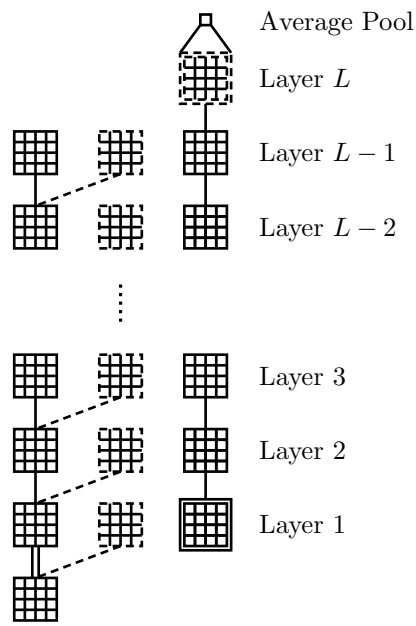


Figure 3.2 Initialization of deep CNN with 4×4 input. The 3 grids per row represent the 3 channels per layer, and the box at the top represents the scalar output of the final average pool. Line styles indicate types of weight initializations (solid:diag$(0, 1, 0)$, double:diag$(0, C_L, 0)$, dash:Gaussian, none:$0_{3\times3}$). Box styles indicate types of bias initializations (solid:0, dash:Gaussian, double:C_L, double-dash:$-C_L$).

4 NTK analysis of infinitely deep CNN

Consider an L-layer convolutional neural network (CNN) $f_\theta^{L+1}\colon \mathbb{R}_+^{d\times d} \to \mathbb{R}$ parameterized by θ, where the input $x \in \mathbb{R}_+^{d\times d}$ is a $d \times d$ image with positive entries and the output is a scalar. The network consists of $L-1$ convolutional layers using 3×3 filters and zero-padding of 1 with 3 output channels, each followed by the ReLU activation function. The L-th convolutional layer uses a 3×3 filter with zero-padding of 1 and has a single output channel. This is followed by a global average pool with no activation function applied before or after the average pool. All convolutional layers use stride of 1.

Let us set up specific notation. For a convolutional filter $w \in \mathbb{R}^{3\times 3}$ and a (single-channel) image $x \in \mathbb{R}_+^{d\times d}$, denote the convolution operation with zero padding as

$$[w * x]_{i,j} = \big\langle w, \phi_{i,j}(x) \big\rangle$$

for $1 \le i, j \le d$, where $\phi_{i,j}(x) = [x]_{i-1:i+1,j-1:j+1}$ and $x_{pq} = 0$ if p or q is less than 1 or greater than d, i.e., $x_{pq} = 0$ if the index is out of bounds. Define $\boldsymbol{\iota}_{3\times 3} = \mathrm{diag}(0,1,0)$ to be the 3×3 filter serving as the identity map. So $\boldsymbol{\iota}_{3\times 3} * x = x$. Define the pre-activation values as

$$\begin{aligned}
\Big(f^1_{\theta^{(1)}}\Big)_{r,:,:} &= w^1_{r,1,:,:} * x + 1_{d\times d} b^1_r, \\
\Big(f^l_{\theta^{(l)}}\Big)_{r,:,:} &= \sum_{s=1}^{n_{l-1}} w^l_{r,s,:,:} * \sigma\left(\Big(f^{l-1}_{\theta^{(l-1)}}\Big)_{s,:,:}\right) + 1_{d\times d} b^l_r,
\end{aligned}$$

for $2 \leq l \leq L$, where $1_{d\times d}$ is the d by d matrix with all unit entries, n_l is the number of channels of the l-th layer, and $r = 1, \ldots, n_l$. Our notation indexing the 3D and 4D tensors resembles the PyTorch convention and is defined precisely in Appendix. We use ReLU for the activation σ. The number of channels are $3 = n_1 = \cdots = n_{L-1}$ and $n_0 = n_L = 1$. So, $f^l_{\theta^{(l)}}(x) \in \mathbb{R}^{n_l \times d \times d}$, $w^l \in \mathbb{R}^{n_l \times n_{l-1} \times 3 \times 3}$, and $b^l \in \mathbb{R}^{n_l}$ for $1 \leq l \leq L$. Write average pool as $S(A) = \frac{1}{d^2}\sum_{i=1}^{d}\sum_{j=1}^{d} A_{i,j}$ for $A \in \mathbb{R}^{d\times d}$. The CNN outputs

$$f^{L+1}_\theta = S(f^L_{\theta^{(L)}}).$$

For $1 \leq l \leq L$, write $\theta^{(l)} = \{w^i, b^i : i \leq l\}$ to denote the collection of parameters up to l-th layer.

4.0.1 Initialization

Initialize the filters of our L-layer CNN f^{L+1}_θ as follows:

$$w^1_{1,1,:,:} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & C_L & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad w^1_{2,1,:,:} = \begin{bmatrix} u^1_{1,1} & u^1_{1,2} & u^1_{1,3} \\ u^1_{2,1} & u^1_{2,2} & u^1_{2,3} \\ u^1_{3,1} & u^1_{3,2} & u^1_{3,3} \end{bmatrix},$$

$$w^1_{3,1,:,:} = 0_{3\times 3}$$

$$w^l_{1,:,:,:} = \boldsymbol{\iota}_{3\times 3}, 0_{3\times 3}, 0_{3\times 3}$$

$$w^l_{2,:,:,:} = \begin{bmatrix} u^l_{1,1} & u^l_{1,2} & u^l_{1,3} \\ u^l_{2,1} & u^l_{2,2} & u^l_{2,3} \\ u^l_{3,1} & u^l_{3,2} & u^l_{3,3} \end{bmatrix}, 0_{3\times 3}, 0_{3\times 3}$$

$$w^l_{3,:,:,:} = 0_{3\times 3}, 0_{3\times 3}, \boldsymbol{\iota}_{3\times 3}$$

$$w^L_{1,:,:,:} = 0_{3\times 3}, 0_{3\times 3}, \boldsymbol{\iota}_{3\times 3}$$

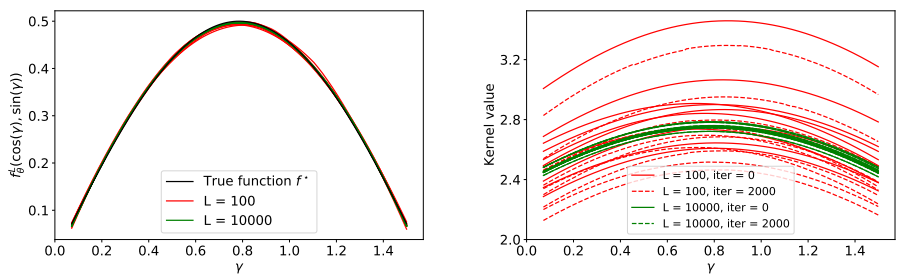
for $2 \leq l \leq L-1$, where $u_{i,j}^l \in \mathbb{R}^{3\times 3}$ is randomly sampled $u_{i,j}^l \overset{iid}{\sim} \mathcal{N}(0, \rho^2)$ for $1 \leq l \leq L-1$. Here, $0_{3\times 3}$ is the 3 by 3 matrix with all zero entries, $C_L > 0$ is a scalar growing as a function of L at a rate satisfying $L^2/C_L \to 0$, and $\rho > 0$ is a fixed variance parameter. Initialize the biases as follows:

$$(b_1^1, b_2^1, b_3^1) = (0, v^1, C_L)$$

$$(b_1^l, b_2^l, b_3^l) = (0, v^l, 0)$$

$$b^L = -C_L$$

for $2 \leq l \leq L-1$, where $v^l \in \mathbb{R}$ is randomly sampled $v^l \overset{iid}{\sim} \mathcal{N}(0, C_L^2\beta^2)$ for $1 \leq l \leq L-1$. Here, $\beta > 0$ is a fixed variance parameter.



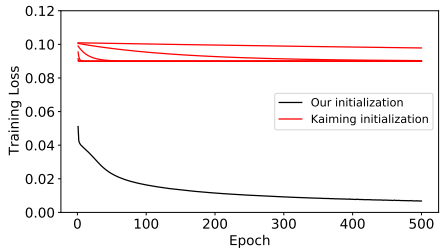
(a) True function and trained MLP after 2000 iterations.

(b) Scaled NTK values

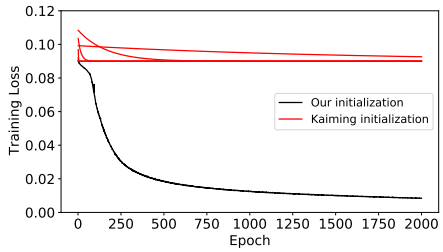
Figure 4.1 Depth L MLPs learning a toy function with 2D inputs as described in Section 5.1, (Left) Trained deep MLP approximates the true function well, i.e., training succeeds. (Right) Kernel values evaluated at initialization and after training with 10 independent initialization-training trials each for $L = 100$ and $L = 10000$. As L grows, initialization becomes less random, as Theorem 3.1 predicts, and the kernel changes less throughout training, as Theorem 3.2 predicts.

Task	Architecture	Depth	C_L	ρ	β	Learning rate	Epochs	Training loss	Test accuracy
10-class	MLP	4000	4	1	1	1×10^{-5}	1500	0.0055	97.48%
10-class	CNN	4000	4	$1/\sqrt{3}$	1	1×10^{-5}	2000	0.014578	94.02%
Binary	CNN	20000	20	$1/\sqrt{3}$	1	1×10^{-8}	1000	0.031257	98.87%

Table 4.1 Very deep MLP and CNNs trained with MNIST. As the theory predicts, the very deep networks are trainable.



(a) Depth 1000 MLP



(b) Depth 1000 CNN

Figure 4.2 Depth 1000 MLPs and CNNs with MNIST are trainable with our proposed initialization but not with the standard Kaiming He initialization. For Kaiming initialization, we show trials with learning rates 1×10^{-5}, 1×10^{-4}, 1×10^{-3}, 0.01, 0.1, and 1.

4.0.2 Convergence in infinite-depth limit

We now analyze the convergence of the infinitely deep CNN.

Theorem 4.1 (Scaled NTK at initialization) *Suppose f_θ^{L+1} is initialized as in Section 4.0.1. For any $x, x' \in \mathbb{R}_+^{d \times d}$,*

$$S\left(\tilde{\Theta}_0^L(x, x')\right) \xrightarrow{p} \tilde{\Theta}^\infty(x, x')$$

as $L \to \infty$, where

$$\tilde{\Theta}^\infty(x, x') = \frac{1}{d^2} \sum_{s=1}^{3} \sum_{u=1}^{3} \Bigg(p_d(s, u) + d^2 S(x_{\psi_{s,u}}) S(x'_{\psi_{s,u}}) + \sum_{i,j \in \psi_{s,u}} \sum_{i',j' \in \psi_{s,u}} \mathbb{E}[\sigma(g(\phi_{i,j}(x)))\sigma(g(\phi_{i',j'}(x')))] \Bigg),$$

$g \sim \mathcal{GP}(0, \rho^2 \langle x, x' \rangle + \beta^2)$, $\psi_{s,u}$ is the set of coordinate which satisfy $x_{\psi_{s,u}} = [x]_{s-1:d+s-2, u-1:d+u-2}$ and $x_{pq} = 0, \phi_{mn}(x) = 0$ if the index is out of bounds, and

$$p_d(s, u) = \begin{cases} d^4 & (s, u) = (2, 2), \\ d^2(d-1)^2 & |s - u| = 1, \\ (d-1)^4 & \text{else.} \end{cases}$$

Theorem 4.2 (Invariance of scaled NTK) *Let $T > 0$. Suppose $\int_0^T \left\| \delta|_{f_\theta^L} \right\|_{p^{in}} dt$ is stochastically bounded as $L \to \infty$. Then, for any $x, x' \in \mathbb{R}_+^{d \times d}$,*

$$S\left(\tilde{\Theta}_t^L(x, x')\right) \xrightarrow{p} \tilde{\Theta}^\infty(x, x')$$

uniformly for $t \in [0, T]$ as $L \to \infty$.

Again, define the kernel regression predictor as

$$f_{\text{ntk}}(x) = \left(\tilde{\Theta}^\infty(x, x_1), \ldots, \tilde{\Theta}^\infty(x, x_N)\right) K^{-1} f^\star(X),$$

where $K_{i,j} = \tilde{\Theta}^\infty(x_i, x_j)$ and $[f^\star(X)]_i = f^\star(x_i)$ for $i, j \in \{1, \ldots, N\}$.

Theorem 4.3 (Equivalence between deep CNN and kernel regression) *Let $\mathcal{L}(f) = \frac{1}{2} \|f - f^\star\|_{p^{in}}^2$. Let $\tilde{\Theta}^\infty$ be positive definite. If f_t follows the kernel gradient flow with respect to $\tilde{\Theta}^\infty$, then for any $x \in \mathbb{R}_+^{d \times d}$,*

$$\lim_{t \to \infty} f_t(x) = f_{\text{ntk}}(x).$$

Generalizations. At the expense of slight notational complications, we can generalize our results as follows. We assumed the convolutional filter size is 3×3, but we can use larger filters by assigning a symbol for the filter size and managing the summation indices with care. We assumed the number of input channels and output scalar dimension are 1, but we can have c input channels and k outputs, i.e., $f_\theta^{L+1} \colon \mathbb{R}_+^{c \times d \times d} \to \mathbb{R}^k$, by letting the intermediate layers have $c + 1 + k$ channels. In fact, Section 5.2 presents a 10-class classification of MNIST with a deep CNN using $1 + 1 + 10 = 12$ channels in intermediate layers.

5 Experiments

In this section, we experimentally demonstrate the invariance of the scaled NTK and the trainability of deep neural networks. The code is available at `https://github.com/lthilnklover/deep-narrow-NTK`

5.1 Convergence of the scaled NTK

Our first experiment, inspired by [27], trains L-layer MLPs on 2-dimensional inputs and shows that the network and its scaled NTK converges as the depth L increases. For $L = 100$ and $L = 10000$, we initialize 10 independent MLP instances and train them to approximate $f^\star(x_1, x_2) = x_1 x_2$ using the quadratic loss. To verify that the networks are indeed successfully trained, we compare the trained MLP against the true function $f^\star$ in Figure 4.1(a). We then plot the scaled NTK $\tilde{\Theta}^L(x_0, x)$ for fixed $x_0 = \left(\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}\right)$ and $x = (\cos(\gamma), \sin(\gamma))$ for $0 < \gamma < \pi/2$ in Figure 4.1(b). The kernels are plotted at initialization ($t = 0$), and after 2000 iterations of gradient descent.

5.2 Trainability of the deep narrow neural network

Next, we demonstrate the empirical trainability of the deep narrow networks on the MNIST dataset.

Very deep MLP. We train L-layer MLPs with $d_{\text{in}} = 784$ and $d_{\text{out}} = 10$ using the quadratic loss with one-hot vectors as targets. To establish a point of comparison, we attempt to train a 1000-layer MLP with the typical Kaiming He uniform initialization [23]. We tuned the learning rate via a grid search from 0.00001 to 1.0, but the network was untrainable, as one would expect based on the prior findings of [24, 67, 26]. In contrast, when we use the initialization defined in Section 3.1, the deep MLP was trainable. Figure 4.2(a) reports these results. To push the depth, we also trained a 4000-layer MLP and report the results in Table 4.1. To the best of our knowledge, this 4000-layer MLP holds the record for the deepest trained MLP.

Very deep CNN. We train L-layer CNNs using the quadratic loss with one-hot probability vectors as targets. To reduce the computational cost, we insert a 4×4 average pool before the first layer to reduce the MNIST input size to 7×7. As in the MLP experiment, we attempt to train a 1000-layer CNN with the typical Kaiming He uniform initialization, but the network was untrainable. In contrast, when we use the initialization defined in Section 4.0.1, the deep CNN was trainable. Figure 4.2(b) reports these results. To push the depth, we also trained a 4000-layer CNN and report the results in Table 4.1. To further push the depth, we simplify the problem to binary classification between digits 0 and 1 and use values 0 and 1 as targets. We then trained a 20000-layer CNN and report the results in Table 4.1. This CNN surpasses the 10000-depth CNN of [72] and, to the best of our knowledge, holds the record of the deepest trained CNN.

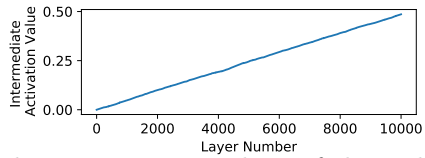


Figure 5.1 Intermediate activation values of the rightmost neurons of each layer after training. (The effect of the bias C_L is compensated for to help visualize the change.)

5.3 Accumulation of the layer-wise effect

To observe the accumulation of the output value throughout the depth, we plot the intermediate activation values of the rightmost neurons of each layer from the 10000-layer MLP trained in Section 5.1. Precisely, we plot $(\sigma(f^{(l)}(x)))_{d_{\text{in}}+d_{\text{out}}+1} - C_L$ for $1 \leq l \leq L-1$ and $(f^{(L)}(x))_{d_{\text{in}}+d_{\text{out}}+1}$. Figure 5.1 shows that the target output value is achieved through the accumulation of the effect of 10000-layers.

6 Conclusion

This work presents an NTK analysis of a deep narrow MLP and CNN in the infinite-depth limit and establishes the first trainability guarantee on deep narrow neural networks. Our results serve as a demonstration that infinitely deep neural networks can be made provably trainable using the right initialization, just as the infinitely wide counterparts are. However, our results do have the following limitations. First, while our proposed initialization is straightforwardly implementable, it is far from the initializations used in practice. Second, our results do not indicate any benefit of using deep neural networks compared to wide neural networks. Further investigating the trainability of overparameterized deep neural networks to address these questions would be an interesting direction of future work.

7 Appendix

As a matter of quantity, we attach a following link to download this paper with appendix: `https://arxiv.org/abs/2202.02981`.

Bibliography

[1] Ben Adlam, Jaehoon Lee, Lechao Xiao, Jeffrey Pennington, and Jasper Snoek: Exploring the uncertainty properties of neural networks' implicit priors in the infinite-width limit. *International Conference on Learning Representations* (2021).

[2] Zeyuan Allen-Zhu, Yuanzhi Li, and Zhao Song: A convergence theory for deep learning via over-parameterization. *International Conference on Machine Learning* (2019).

[3] Sanjeev Arora, Nadav Cohen, Noah Golowich, and Wei Hu: A convergence analysis of gradient descent for deep linear neural networks. *International Conference on Learning Representations* (2019).

[4] Sanjeev Arora, Simon S Du, Wei Hu, Zhiyuan Li, Russ R Salakhutdinov, and Ruosong Wang: On exact computation with an infinitely wide neural net. *Neural Information Processing Systems* (2019).

[5] A.R. Barron: Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Transactions on Information Theory*, **39** (1993), 930–945.

[6] Alberto Bietti, and Julien Mairal: On the inductive bias of neural tangent kernels. *Neural Information Processing Systems* (2019).

[7] Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud: Neural ordinary differential equations. *Neural Information Processing Systems* (2018).

[8] Zixiang Chen, Yuan Cao, Quanquan Gu, and Tong Zhang: A generalized neural tangent kernel analysis for two-layer neural networks. *Neural Information Processing Systems* (2020).

[9] Lénaïc Chizat, and Francis Bach: On the global convergence of gradient descent for over-parameterized models using optimal transport. *Neural Information Processing Systems* (2018).

[10] Lénaïc Chizat, Edouard Oyallon, and Francis R. Bach: On lazy training in differentiable programming. *Neural Information Processing Systems* (2019).

[11] Youngmin Cho, and Lawrence Saul: Kernel methods for deep learning. *Neural Information Processing Systems* (2009).

[12] G. Cybenko: Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, **2** (1989), 303–314.

[13] Zhiyan Ding, Shi Chen, Qin Li, and Stephen Wright: On the global convergence of gradient descent for multi-layer ResNets in the mean-field regime. *arXiv:2110.02926* (2021). arXiv: `2110.02926`.

[14] Zhiyan Ding, Shi Chen, Qin Li, and Stephen J Wright: Overparameterization of deep resnet: zero loss and mean-field analysis. *Journal of Machine Learning Research*, **23** (2022), 1–65.

[15] Simon S Du, Jason Lee, Haochuan Li, Liwei Wang, and Xiyu Zhai: Gradient descent finds global minima of deep neural networks. *International Conference on Machine Learning* (2019).

[16] Simon S. Du, Xiyu Zhai, Barnabás Póczos, and Aarti Singh: Gradient descent provably optimizes over-parameterized neural networks. *International Conference on Learning Representations* (2019).

[17] Ken-Ichi Funahashi: On the approximate realization of continuous mappings by neural networks. *Neural Networks*, **2** (1989), 183–192.

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

[19] Boris Hanin: Universal function approximation by deep neural nets with bounded width and ReLU activations. *Mathematics*, **7** (2019), 992.

[20] Boris Hanin: Which neural net architectures give rise to exploding and vanishing gradients? *Neural Information Processing Systems* (2018).

[21] Boris Hanin, and Mihai Nica: Finite depth and width corrections to the neural tangent kernel. *International Conference on Learning Representations* (2020).

[22] Boris Hanin, and Mark Sellke: Approximating continuous functions by ReLU nets of minimal width. *arXiv:1710.11278v2* (2017). arXiv: `1710.11278v2`.

[23] K. He, X. Zhang, S. Ren, and J. Sun: Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification. *International Conference on Computer Vision* (2015).

[24] Kaiming He, and Jian Sun: Convolutional neural networks at constrained time cost. *Computer Vision and Pattern Recognition* (2015).

[25] Kurt Hornik: Approximation capabilities of multilayer feedforward networks. *Neural Networks*, **4** (1991), 251–257.

[26] Kaixuan Huang, Yuqing Wang, Molei Tao, and Tuo Zhao: Why do deep residual networks generalize better than deep feedforward networks? — A neural tangent kernel perspective. *Neural Information Processing Systems* (2020).

[27] Arthur Jacot, Franck Gabriel, and Clement Hongler: Neural tangent kernel: Convergence and generalization in neural networks. *Neural Information Processing Systems* (2018).

[28] Ziwei Ji, Matus Telgarsky, and Ruicheng Xian: Neural tangent kernels, transportation mappings, and universal approximation. *International Conference on Learning Representations* (2020).

[29] Jesse Johnson: Deep, skinny neural networks are not universal approximators. *International Conference on Learning Representations* (2019).

[30] Lee K. Jones: A simple lemma on greedy approximation in hilbert space and convergence rates for projection pursuit regression and neural network training. *The Annals of Statistics*, **20** (1992), 608–613.

[31] Patrick Kidger, and Terry Lyons: Universal approximation with deep narrow networks. *Conference on Learning Theory* (2020).

[32] Jaehoon Lee, Yasaman Bahri, Roman Novak, Samuel S. Schoenholz, Jeffrey Pennington, and Jascha Sohl-Dickstein: Deep neural networks as gaussian processes. *International Conference on Learning Representations* (2018).

[33] Jaehoon Lee, Samuel Schoenholz, Jeffrey Pennington, Ben Adlam, Lechao Xiao, Roman Novak, and Jascha Sohl-Dickstein: Finite versus infinite neural networks: An empirical study. *Neural Information Processing Systems* (2020).

[34] Jaehoon Lee, Lechao Xiao, Samuel Schoenholz, Yasaman Bahri, Roman Novak, Jascha Sohl-Dickstein, and Jeffrey Pennington: Wide neural networks of any depth evolve as linear models under gradient descent. *Neural Information Processing Systems* (2019).

[35] Moshe Leshno, Vladimir Ya. Lin, Allan Pinkus, and Shimon Schocken: Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. *Neural Networks*, **6** (1993), 861–867.

[36] Yuanzhi Li, and Yingyu Liang: Learning overparameterized neural networks via stochastic gradient descent on structured data. *Neural Information Processing Systems* (2018).

[37] Hongzhou Lin, and Stefanie Jegelka: ResNet with one-neuron hidden layers is a universal approximator. *Neural Information Processing Systems* (2018).

[38] Jianfeng Lu, Zuowei Shen, Haizhao Yang, and Shijun Zhang: Deep network approximation for smooth functions. *SIAM Journal on Mathematical Analysis*, **53** (2021), 5465–5506. eprint: `https://doi.org/10.1137/20M134695X`.

[39] Yiping Lu, Chao Ma, Yulong Lu, Jianfeng Lu, and Lexing Ying: A mean field analysis of deep resnet and beyond: Towards provably optimization via overparameterization from depth. *International Conference on Machine Learning* (2020).

[40] Zhou Lu, Hongming Pu, Feicheng Wang, Zhiqiang Hu, and Liwei Wang: The expressive power of neural networks: A view from the width. *Neural Information Processing Systems* (2017).

[41] Alexander G. de G. Matthews, Jiri Hron, Mark Rowland, Richard E. Turner, and Zoubin Ghahramani: Gaussian process behaviour in wide deep neural networks. *International Conference on Learning Representations* (2018).

[42] 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** (2018), E7665–E7671. eprint: `https://www.pnas.org/content/115/33/E7665.full.pdf`.

[43] Radford M Neal: *Bayesian Learning for Neural Networks*. University of Toronto, 1996.

[44] Radford M. Neal: *Priors for Infinite Networks*. Tech. rep. crg-tr-94-1. University of Toronto, 1994.

[45] Phan-Minh Nguyen, and Huy Tuan Pham: A rigorous framework for the mean field limit of multilayer neural networks. *arXiv:2001.11443* (2021). arXiv: `2001.11443`.

[46] Timothy Nguyen, Roman Novak, Lechao Xiao, and Jaehoon Lee: Dataset distillation with infinitely wide convolutional networks. *Neural Information Processing Systems* (2021).

[47] Roman Novak, Lechao Xiao, Yasaman Bahri, Jaehoon Lee, Greg Yang, Jiri Hron, Daniel A. Abolafia, Jeffrey Pennington, and Jascha Sohl-dickstein: Bayesian deep convolutional networks with many channels are gaussian processes. *International Conference on Learning Representations* (2019).

[48] Roman Novak, Lechao Xiao, Jiri Hron, Jaehoon Lee, Alexander A. Alemi, Jascha Sohl-Dickstein, and Samuel S. Schoenholz: Neural tangents: Fast and easy infinite neural networks in python. *International Conference on Learning Representations* (2020).

[49] Sejun Park, Chulhee Yun, Jaeho Lee, and Jinwoo Shin: Minimum width for universal approximation. *International Conference on Learning Representations* (2021).

[50] Stefano Peluchetti, and Stefano Favaro: Doubly infinite residual neural networks: a diffusion process approach. *Journal of Machine Learning Research* (2021).

[51] Stefano Peluchetti, and Stefano Favaro: Infinitely deep neural networks as diffusion processes. *International Conference on Artificial Intelligence and Statistics* (2020).

[52] Jeffrey Pennington, Samuel Schoenholz, and Surya Ganguli: Resurrecting the sigmoid in deep learning through dynamical isometry: Theory and practice. *Neural Information Processing Systems* (2017).

[53] Huy Tuan Pham, and Phan-Minh Nguyen: Global convergence of three-layer neural networks in the nean field regime. *International Conference on Learning Representations* (2021).

[54] Allan Pinkus: Approximation theory of the MLP model in neural networks. *Acta Numerica*, **8** (1999), 143–195.

[55] Ali Rahimi, and Benjamin Recht: Random features for large-scale kernel machines. *Neural Information Processing Systems* (2007).

[56] Ali Rahimi, and Benjamin Recht: Uniform approximation of functions with random bases. *Allerton Conference on Communication, Control, and Computing* (2008).

[57] Ali Rahimi, and Benjamin Recht: Weighted sums of random kitchen sinks: Replacing minimization with randomization in learning. *Neural Information Processing Systems* (2008).

[58] Carl Edward Rasmussen: Gaussian Processes in Machine Learning. *Advanced Lectures on Machine Learning*. Ed. by Olivier Bousquet, Ulrike von Luxburg, and Gunnar Rätsch. Springer, 2004, 63–71.

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

[60] Grant Rotskoff, Samy Jelassi, Joan Bruna, and Eric Vanden-Eijnden: Neuron birth-death dynamics accelerates gradient descent and converges asymptotically. *International Conference on Machine Learning* (2019).

[61] Grant Rotskoff, and Eric Vanden-Eijnden: Parameters as interacting particles: Long time convergence and asymptotic error scaling of neural networks. *Neural Information Processing Systems* (2018).

[62] Ohad Shamir: Exponential convergence time of gradient descent for one-dimensional deep linear neural networks. *Conference on Learning Theory* (2019).

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

[64] Justin Sirignano, and Konstantinos Spiliopoulos: Mean field analysis of neural networks: A law of large numbers. *SIAM Journal on Applied Mathematics*, **80** (2020), 725–752.

[65] Jascha Sohl-Dickstein, Roman Novak, Samuel S. Schoenholz, and Jaehoon Lee: On the infinite width limit of neural networks with a standard parameterization. *arXiv:2001.07301* (2020). arXiv: `2001.07301`.

[66] Mahdi Soltanolkotabi, Adel Javanmard, and Jason D. Lee: Theoretical insights into the optimization landscape of over-parameterized shallow neural networks. *IEEE Transactions on Information Theory*, **65** (2019), 742–769.

[67] Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber: Highway networks. *arXiv:1505.00387* (2015). arXiv: `1505.00387`.

[68] Paulo Tabuada, and Bahman Gharesifard: Universal approximation power of deep residual neural networks via nonlinear control theory. *International Conference on Learning Representations* (2021).

[69] Matus Telgarsky: Benefits of depth in neural networks. *Conference on Learning Theory* (2016).

[70] Christopher Williams: Computing with infinite networks. *Neural Information Processing Systems* (1997). Ed. by M. C. Mozer, M. Jordan, and T. Petsche.

[71] 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. *Conference on Learning Theory* (2020).

[72] Lechao Xiao, Yasaman Bahri, Jascha Sohl-Dickstein, Samuel Schoenholz, and Jeffrey Pennington: Dynamical isometry and a mean field theory of CNNs: How to train 10,000-layer vanilla convolutional neural networks. *International Conference on Machine Learning* (2018).

[73] Greg Yang: Scaling limits of wide neural networks with weight sharing: Gaussian process behavior, gradient independence, and neural tangent kernel derivation. *arXiv:1902.04760v3* (2019). arXiv: `1902.04760v3`.

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

[75] Difan Zou, Yuan Cao, Dongruo Zhou, and Quanquan Gu: Gradient descent optimizes over-parameterized deep ReLU networks. *Machine Learning*, **109** (2020), 467–492.

초 록

과매개화된 신경망의 훈련 역학을 분석하는 최근의 엄청난 발전은 주로 넓은 네트워크에 초점을 맞추었기 때문에 딥 러닝에서 깊이의 역할을 충분히 다루지 못한다. 이 논문에서 우리는 무한히 깊지만 좁은 신경망의 훈련 가능성을 처음으로 보인다. 우리는 특정 초기화하에서 무한한 깊이의 다층 신경망을 연구하고 뉴럴 탄젠트커널 이론을 사용하여 학습 가능성을 보장한다. 그런 다음 분석을 무한히 깊은 합성곱 신경망으로 확장하고 간단한 실험을 수행한다.

주요어: 딥러닝, 뉴럴탄젠트커널 이론, 다층신경망

학 번: 2020-20857

감사의 글

석사 심사를 봐주신 박형빈 교수님, 오토 교수님, 그리고 지도 교수님이신 류경석 교수님에게 감사의 말씀을 드립니다. 또 공동 저자인 노승문 교수님과 최주영 학생에게도 감사의 뜻을 전하며 논문 작성과정에서 도움을 준 박지선 학생과 윤태호 학생에게도 감사하다는 말을 전합니다.