Dynamics of Deep Neural Networks and Neural Tangent Hierarchy

Jiaoyang Huang [*1] **Horng-Tzer Yau** [*2]

Abstract

The evolution of a deep neural network trained by the gradient descent in the overparametrization regime can be described by its neural tangent kernel (NTK) (Jacot et al., 2018; Du et al., 2018b;a; Arora et al., 2019b). It was observed (Arora et al., 2019a) that there is a performance gap between the kernel regression using the limiting NTK and the deep neural networks. We study the dynamic of neural networks of finite width and derive an infinite hierarchy of differential equations, the neural tangent hierarchy (NTH). We prove that the NTH hierarchy truncated at the level $p \geqslant 2$ approximates the dynamic of the NTK up to arbitrary precision under certain conditions on the neural network width and the data set dimension. The assumptions needed for these approximations become weaker as p increases. Finally, NTH can be viewed as higher order extensions of NTK. In particular, the NTH truncated at $p = 2$ recovers the NTK dynamics.

1. Introduction

Deep neural networks have become popular due to their unprecedented success in a variety of machine learning tasks. Image recognition (LeCun et al., 1998; Krizhevsky et al., 2012; Szegedy et al., 2015), speech recognition (Hinton et al., 2012; Sainath et al., 2013), playing Go (Silver et al., 2016; 2017) and natural language understanding (Collobert et al., 2011; Wu et al., 2016; Devlin et al., 2018) are just a few of the recent achievements. However, one aspect of deep neural networks that is not well understood is training. Training a deep neural network is usually done via a gradient decent based algorithm. Analyzing such training dynamics is challenging. Firstly, as highly nonlinear structures, deep neural networks usually involve a large number of parameters. Secondly, as highly non-convex optimization problems, there is no guarantee that a gradient based algorithm will be able to find the optimal parameters efficiently during the training of neural networks. *One question then arises: given such complexities, is it possible to obtain a succinct description of the training dynamics?*

In this paper, we focus on the empirical risk minimization problem with the quadratic loss function

$$\min_{\theta} L(\theta) = \frac{1}{2n} \sum_{\alpha=1}^{n} (f(x_\alpha, \theta) - y_\alpha)^2$$

where $\{x_\alpha\}_{\alpha=1}^n$ are the training inputs, $\{y_\alpha\}_{\alpha=1}^n$ are the labels, and the dependence is modeled by a deep fully-connected feedforward neural network with H hidden layers. The network has d input nodes, and the input vector is given by $x \in \mathbb{R}^d$. For $1 \leqslant \ell \leqslant H$, the ℓ-th hidden layer has m neurons. Let $x^{(\ell)}$ be the output of the ℓ-th layer with $x^{(0)} = x$. Then the feedforward neural network is given by the set of recursive equations:

$$x^{(\ell)} = \frac{1}{\sqrt{m}} \sigma(W^{(\ell)} x^{(\ell-1)}), \quad \ell = 1, 2, \cdots, H, \tag{1}$$

where $W^{(\ell)} \in \mathbb{R}^{m \times d}$ if $\ell = 1$ and $W^{(\ell)} \in \mathbb{R}^{m \times m}$ if $2 \leqslant \ell \leqslant H$ are the weight matrices, and σ is the activation unit, which is applied coordinate-wise to its input. The output of the neural network is

$$f(x, \theta) = a^\top x^{(H)} \in \mathbb{R}, \tag{2}$$

where $a \in \mathbb{R}^m$ is the weight matrix for the output layer. We denote the vector containing all trainable parameters by $\theta = (\mathrm{vec}(W^{(1)}), \mathrm{vec}(W^{(2)}) \ldots, \mathrm{vec}(W^{(H)}), a)$. We remark that this parametrization is nonstandard because of those $1/\sqrt{m}$ factors. However, it has already been adopted in several recent works (Jacot et al., 2018; Du et al., 2018b;a; Lee et al., 2019). We note that the predictions and training dynamics of (1) are identical to those of standard networks, up to a scaling factor $1/\sqrt{m}$ in the learning rate for each parameter.

We initialize the neural network with random Gaussian weights following the Xavier initialization scheme (Glorot & Bengio, 2010). More precisely, we set the initial parameter vector θ_0 as $W_{ij}^{(\ell)} \sim \mathcal{N}(0, \sigma_w^2)$, $a_i \sim \mathcal{N}(0, \sigma_a^2)$. In this way, for the randomly initialized neural network,

[*]Equal contribution [1]School of Mathematics, IAS, Princeton, NJ, USA [2]Mathematics Department, Harvard, Cambridge, MA, USA. Correspondence to: Jiaoyang Huang <jiaoyang@ias.edu>.

Proceedings of the 37^{th} International Conference on Machine Learning, Vienna, Austria, PMLR 119, 2020.

we have that the L_2 norms of the output of each layer are of order one, i.e. $\|x^{(\ell)}\|_2^2 = \mathrm{O}(1)$ for $0 \leqslant \ell \leqslant H$, and $f(x, \theta_0) = \mathrm{O}(1)$ with high probability. In this paper, we train all layers of the neural network with continuous time gradient descent (gradient flow): for any time $t \geqslant 0$

$$\begin{aligned} &\partial_t W_t^{(\ell)} = -\partial_{W^{(\ell)}} L(\theta_t), \quad \ell = 1, 2, \cdots, H, \\ &\partial_t a_t = -\partial_a L(\theta_t), \end{aligned} \tag{3}$$

where $\theta_t = (\mathrm{vec}(W_t^{(1)}), \mathrm{vec}(W_t^{(2)}) \ldots, \mathrm{vec}(W_t^{(H)}), a_t)$.

For simplicity of notations, we write $\sigma(W^{(\ell)} x^{(\ell-1)})$ as $\sigma_\ell(x)$, or simply σ_ℓ if the context is clear. We write its derivative $\mathrm{diag}(\sigma'(W^{(\ell)} x^{(\ell-1)}))$ as $\sigma'_\ell(x) = \sigma_\ell^{(1)}(x)$, and r-th derivative $\mathrm{diag}(\sigma^{(r)}(W^{(\ell)} x^{(\ell-1)}))$ as $\sigma_\ell^{(r)}(x)$, or $\sigma_\ell^{(r)}$ for $r \geqslant 1$. In this notation, $\sigma_\ell^{(r)}(x)$ are diagonal matrices. With those notations, explicitly, the continuous time gradient descent dynamic (3) is

$$\begin{aligned} &\partial_t W_t^{(\ell)} = -\partial_{W^{(\ell)}} L(\theta_t) \\ &= -\frac{1}{n} \sum_{\beta=1}^{n} \left(\sigma'_\ell(x_\beta) \frac{(W_t^{(\ell+1)})^\top}{\sqrt{m}} \cdots \sigma'_H(x_\beta) \frac{a_t}{\sqrt{m}} \right) \\ &\quad \otimes (x_\beta^{(\ell-1)})^\top (f(x_\beta, \theta_t) - y_\beta), \end{aligned} \tag{4}$$

for $\ell = 1, 2, \cdots, H$, and

$$\partial_t a_t = -\partial_a L(\theta_t) = -\frac{1}{n} \sum_{\beta=1}^{n} x_\beta^{(H)} (f(x_\beta, \theta_t) - y_\beta). \tag{5}$$

1.1. Neural Tangent Kernel

A recent paper (Jacot et al., 2018) introduced the Neural Tangent Kernel (NTK) and proved the limiting NTK captures the behavior of fully-connected deep neural networks in the infinite width limit trained by gradient descent:

$$\begin{aligned} &\partial_t f(x, \theta_t) = \partial_\theta f(x, \theta_t) \partial_t \theta_t = -\partial_\theta f(x, \theta_t) \partial_\theta L(\theta_t) \\ &= -\frac{1}{n} \partial_\theta f(x, \theta_t) \sum_{\beta=1}^{n} \partial_\theta f(x_\beta, \theta_t) (f(x_\beta, \theta_t) - y_\beta) \\ &= -\frac{1}{n} \sum_{\beta=1}^{n} K_t^{(2)}(x, x_\beta) (f(x_\beta, \theta_t) - y_\beta), \end{aligned} \tag{6}$$

where the NTK $K_t^{(2)}(\cdot, \cdot)$ is given by

$$\begin{aligned} K_t^{(2)}(x_\alpha, x_\beta) &= \langle \partial_\theta f(x_\alpha, \theta_t), \partial_\theta f(x_\beta, \theta_t) \rangle \\ &= \sum_{\ell=1}^{H+1} G_t^{(\ell)}(x_\alpha, x_\beta), \end{aligned} \tag{7}$$

and for $1 \leqslant \ell \leqslant H$,

$$\begin{aligned} &G_t^{(\ell)}(x_\alpha, x_\beta) = \langle \partial_{W^{(\ell)}} f(x_\alpha, \theta_t), \partial_{W^{(\ell)}} f(x_\beta, \theta_t) \rangle \\ &= \left\langle \sigma'_\ell(x_\alpha) \frac{(W_t^{(\ell+1)})^\top}{\sqrt{m}} \cdots \sigma'_H(x_\alpha) \frac{a_t}{\sqrt{m}}, \right. \\ &\left. \sigma'_\ell(x_\beta) \frac{(W_t^{(\ell+1)})^\top}{\sqrt{m}} \cdots \sigma'_H(x_\beta) \frac{a_t}{\sqrt{m}} \right\rangle \langle x_\alpha^{(\ell-1)}, x_\beta^{(\ell-1)} \rangle, \end{aligned}$$

and

$$G_t^{(H+1)} = \langle \partial_a f(x_\alpha, \theta_t), \partial_a f(x_\beta, \theta_t) \rangle = \langle x_\alpha^{(H)}, x_\beta^{(H)} \rangle.$$

The NTK $K_t^{(2)}(\cdot, \cdot)$ varies along training. However, in the infinite width limit, the training dynamic is very simple: The NTK does not change along training, $K_t^{(2)}(\cdot, \cdot) = K_\infty^{(2)}(\cdot, \cdot)$. The network function $f(x, \theta_t)$ follows a linear differential equation (Jacot et al., 2018):

$$\partial_t f(x, \theta_t) = -\frac{1}{n} \sum_{\beta=1}^{n} K_\infty^{(2)}(x, x_\beta) (f(x_\beta, \theta_t) - y_\beta), \tag{8}$$

which becomes analytically tractable. In other words, the training dynamic is equivalent to the kernel regression using the limiting NTK $K_\infty^{(2)}(\cdot, \cdot)$. While the linearization (8) is only exact in the infinite width limit, for a sufficiently wide deep neural network, (8) still provides a good approximation of the learning dynamic for the corresponding deep neural network (Du et al., 2018b;a; Lee et al., 2019). As a consequence, it was proven in (Du et al., 2018b;a) that, for a fully-connected wide neural network with $m \gtrsim n^4$ under certain assumptions on the data set, the gradient descent converges to zero training loss at a linear rate. Although highly overparametrized neural networks is equivalent to the kernel regression, it is possible to show that the class of finite width neural networks is more expressive than the limiting NTK. It has been constructed in (Ghorbani et al., 2019; Yehudai & Shamir, 2019; Allen-Zhu & Li, 2019) that there are simple functions that can be efficiently learnt by finite width neural networks, but not the kernel regression using the limiting NTK.

1.2. Contribution

There is a performance gap between the kernel regression (8) using the limiting NTK and the deep neural networks. It was observed in (Arora et al., 2019a) that the convolutional neural networks outperform their corresponding limiting NTK by 5% - 6%. This performance gap is likely to originate from the change of the NTK along training due to the finite width effect. The change of the NTK along training has its benefits on generalization.

In the current paper, we study the dynamic of the NTK for finite width deep fully-connected neural networks. Here we summarize our main contributions:

- We show the gradient descent dynamic is captured by an infinite hierarchy of ordinary differential equations, the neural tangent hierarchy (NTH). Similar recursive differential equations were also obtained by Dyer and Gur-Ari (Dyer & Gur-Ari, 2019). Different from the limiting NTK (7), which depends only on the neural network architecture, the NTH is data dependent and capable of learning data-dependent features.

- We derive a priori estimates of the higher order kernels involved in the NTH. Using these a priori estimates as input, we confirm a numerical observation in (Lee et al., 2019) that the NTK varies at a rate of order $\mathrm{O}(1/m)$. As a corollary, this implies that for a fully-connected wide neural network with $m \gtrsim n^3$, the gradient descent converges to zero training loss at a linear rate, which improves the results in (Du et al., 2018a).

- The NTH is just an infinite sequence of relationship. Without truncation, it cannot be used to determine the dynamic of the NTK. Using the a priori estimates of the higher order kernels as input, we construct a truncated hierarchy of ordinary differential equations, the truncated NTH. We show that this system of truncated equations approximates the dynamic of the NTK to certain time up to arbitrary precision. This description makes it possible to directly study the change of the NTK for deep neural networks.

1.3. Notations

In the paper, we fix a large constant $p^* > 0$, which appears in Assumptions (2.1) and (2.2). We use $\mathtt{c}, \mathtt{C}$ to represent universal constants, which might be different from line to line. In the paper, we write $a = \mathrm{O}(b)$ or $a \lesssim b$ if there exists some large universal constant $\mathtt{C}$ such that $|a| \leqslant \mathtt{C}b$. We write $a \gtrsim b$ if there exists some small universal constant $\mathtt{c} > 0$ such that $a \geqslant \mathtt{c}b$. We write $a \asymp b$ if there exist universal constants $\mathtt{c}, \mathtt{C}$ such that $\mathtt{c}b \leqslant |a| \leqslant \mathtt{C}b$. We reserve n for the number of input samples and m for the width of the neural network. For practical neural networks, we always have that $m \lesssim \mathrm{poly}(n)$ and $n \lesssim \mathrm{poly}(m)$. We denote the set of input samples as $\mathcal{X} = \{x_1, x_2, \cdots, x_n\}$. For simplicity of notations, we write the output of the neural network as $f_\beta(t) = f(x_\beta, \theta_t)$. We denote vector L_2 norm as $\|\cdot\|_2$, vector or function L_∞ norm as $\|\cdot\|_\infty$, matrix spectral norm as $\|\cdot\|_{2\to 2}$, and matrix Frobenius norm as $\|\cdot\|_{\mathrm{F}}$. We say that an event holds with high probability, if it holds with probability at least $1 - e^{-m^{\mathtt{c}}}$ for some $\mathtt{c} > 0$. Then the intersection of $\mathrm{poly}(n, m)$ many high probability events is still a high probability event, provided m is large enough. In the paper, we treat $\mathtt{c}_r, \mathtt{C}_r$ in Assumption 2.1 and 2.2, and the depth H as constants. We will not keep track of them.

1.4. Related Work

In this section, we survey an incomplete list of previous works on optimization aspect of deep neural networks.

Because of the highly non-convexity nature of deep neural networks, the gradient based algorithms can potentially get stuck near a critical point, i.e., saddle point or local minimum. So one important question in deep neural networks is: what does the loss landscape look like. One promising candidate for loss landscapes is the class of functions that satisfy: (i) all local minima are global minima and (ii) there exists a negative curvature for every saddle point. A line of recent results show that, in many optimization problems of interest (Ge et al., 2015; 2016; Sun et al., 2018; 2016; Bhojanapalli et al., 2016; Park et al., 2016), loss landscapes are in such class. For this function class, (perturbed) gradient descent (Jin et al., 2017; Ge et al., 2015; Lee et al., 2016) can find a global minimum. However, even for a three-layer linear network, there exists a saddle point that does not have a negative curvature (Kawaguchi, 2016). So it is unclear whether this geometry-based approach can be used to obtain the global convergence guarantee of first-order methods. Another approach is to show that practical deep neural networks allow some additional structure or assumption to make non-convex optimizations tractable. Under certain simplification assumptions, it has been proven recently that there are novel loss landscape structures in deep neural networks, which may play a role in making the optimization tractable (Dauphin et al., 2014; Choromanska et al., 2015; Kawaguchi, 2016; Liang et al., 2018; Kawaguchi & Kaelbling, 2019).

Recently, it was proved in a series of papers that, if the size of a neural network is significantly larger than the size of the dataset, the (stochastic) gradient descent algorithm can find optimal parameters (Li & Liang, 2018; Du et al., 2018b; Song & Yang, 2019; Du et al., 2018a; Allen-Zhu et al., 2018b; Zou et al., 2018; Zou & Gu, 2019). In the overparametrization regime, a fully-trained deep neural network is indeed equivalent to the kernel regression predictor using the limiting NTK (8). As a consequence, the gradient descent achieves zero training loss for a deep overparameterized neural network. Under further assumptions, it can be shown that the trained networks generalize (Arora et al., 2019b; Allen-Zhu et al., 2018a; Cao & Gu, 2019). Unfortunately, there is a significant gap between the overparametrized neural networks, which are provably trainable, and neural networks in common practice. Typically, deep neural networks used in practical applications are trainable, and yet, much smaller than what the previous theories require to ensure trainability. In (Kawaguchi & Huang, 2019), it is proven that gradient descent can find a global minimum for certain deep neural networks of sizes commonly encountered in practice.

Training dynamics of neural networks in the mean field setting have been studied in (Mei et al., 2019; Song et al., 2018; Araújo et al., 2019; Nguyen, 2019; Sirignano & Spiliopoulos, 2019; Chizat & Bach, 2018). Their mean field analysis describes distributional dynamics of neural network parameters via certain nonlinear partial differential equations, in the asymptotic regime of large network sizes and large number of stochastic gradient descent training iterations. However, their analysis is restricted to neural networks in the mean-field framework with a normalization factor $1/m$, different from ours $1/\sqrt{m}$, which is commonly used in modern networks (Glorot & Bengio, 2010).

2. Main results

Assumption 2.1. *The activation function σ is smooth, and for any $1 \leqslant r \leqslant 2p^*+1$, there exists a constant $\mathtt{C}_r > 0$ such that the r-th derivative of σ satisfies $\|\sigma^{(r)}(x)\|_\infty \leqslant \mathtt{C}_r$.*

Assumption 2.1 is satisfied by using common activation units such as sigmoid and hyperbolic tangents. Moreover, the softplus activation, which is defined as $\sigma_a(x) = \ln(1+\exp(ax))/a$, satisfies Assumption 2.1 with any hyperparameter $a \in \mathbb{R}_{>0}$. The softplus activation can approximate the ReLU activation for any desired accuracy as

$$\sigma_a(x) \to \mathrm{relu}(x) \text{ as } a \to \infty,$$

where relu represents the ReLU activation.

Assumption 2.2. *There exists a small constant $\mathtt{c} > 0$ such that the training inputs satisfy $\mathtt{c} < \|x_\alpha\|_2 \leqslant \mathtt{c}^{-1}$. For any $1 \leqslant r \leqslant 2p^*+1$, there exists a constant $\mathtt{c}_r > 0$ such that for any distinct indices $1 \leqslant \alpha_1, \alpha_2, \cdots, \alpha_r \leqslant n$, the smallest singular value of the data matrix $[x_{\alpha_1}, x_{\alpha_2}, \cdots, x_{\alpha_r}]$ is at least $\mathtt{c}_r$.*

For more general input data, we can always normalize them such that $\mathtt{c} < \|x_\alpha\|_2 \leqslant \mathtt{c}^{-1}$. Under this normalization, for the randomly initialized deep neural network, it holds that $\|x_\alpha^{(\ell)}\|_2 = \mathrm{O}(1)$ for all $1 \leqslant \ell \leqslant H$, where the implicit constants depend on ℓ. The second part of Assumption 2.2 requires that for any small number of input data: $x_{\alpha_1}, x_{\alpha_2}, \cdots, x_{\alpha_r}$, they are linearly independent.

Theorem 2.3. *Under Assumptions 2.1 and 2.2, there exists an infinite family of operators $K_t^{(r)} : \mathcal{X}^r \mapsto \mathbb{R}$ for $r \geqslant 2$, the continuous time gradient descent dynamic is given by an infinite hierarchy of ordinary differential equations, i.e., the NTH,*

$$\partial_t(f_\alpha(t) - y_\alpha) = -\frac{1}{n}\sum_{\beta=1}^n K_t^{(2)}(x_\alpha, x_\beta)(f_\beta(t) - y_\beta), \tag{9}$$

and for any $r \geqslant 2$,

$$\begin{aligned}&\partial_t K_t^{(r)}(x_{\alpha_1}, x_{\alpha_2}, \cdots, x_{\alpha_r})\\ &= -\frac{1}{n}\sum_{\beta=1}^n K_t^{(r+1)}(x_{\alpha_1}, x_{\alpha_2}, \cdots, x_{\alpha_r}, x_\beta)(f_\beta(t) - y_\beta).\end{aligned} \tag{10}$$

There exists a deterministic family (independent of m) of operators $\mathfrak{K}^{(r)} : \mathcal{X}^r \mapsto \mathbb{R}$ for $2 \leqslant r \leqslant p^*+1$ and $\mathfrak{K}^{(r)} = 0$ if r is odd, such that with high probability with respect to the random initialization, there exist some constants $\mathtt{C}, \mathtt{C}' > 0$ such that

$$\left\| K_0^{(r)} - \frac{\mathfrak{K}^{(r)}}{m^{r/2-1}} \right\|_\infty \lesssim \frac{(\ln m)^{\mathtt{C}}}{m^{(r-1)/2}}, \tag{11}$$

and for $0 \leqslant t \leqslant m^{\frac{p^*}{2(p^*+1)}}/(\ln m)^{\mathtt{C}'}$,

$$\|K_t^{(r)}\|_\infty \lesssim \frac{(\ln m)^{\mathtt{C}}}{m^{r/2-1}}. \tag{12}$$

It was proven in (Du et al., 2018a; Lee et al., 2019) that the change of the NTK for a wide deep neural network is upper bounded by $\mathrm{O}(1/\sqrt{m})$. However, the numerical experiments in (Lee et al., 2019) indicate the change of the NTK is closer to $\mathrm{O}(1/m)$. As a corollary of Theorem 2.3, we confirm the numerical observation that the NTK varies at a rate of order $\mathrm{O}(1/m)$.

Corollary 2.4. *Under Assumptions 2.1 and 2.2, the NTK $K_t^{(2)}(\cdot,\cdot)$ varies at a rate of order $\mathrm{O}(1/m)$: with high probability with respect to the random initialization, there exist some constants $\mathtt{C}, \mathtt{C}' > 0$ such that for $0 \leqslant t \leqslant m^{\frac{p^*}{2(p^*+1)}}/(\ln m)^{\mathtt{C}'}$, it holds*

$$\|\partial_t K_t^{(2)}\|_\infty \lesssim \frac{(1+t)(\ln m)^{\mathtt{C}}}{m}.$$

As another corollary of Theorem 2.3, for a fully-connected wide neural network with $m \gtrsim n^3$, the gradient descent converges to zero training loss at a linear rate.

Corollary 2.5. *Under Assumptions 2.1 and 2.2, we further assume that there exists $\lambda > 0$ (which might depend on n)*

$$\lambda_{\min}\left[K_0^{(2)}(x_\alpha, x_\beta)\right]_{1\leqslant\alpha,\beta\leqslant n} \geqslant \lambda, \tag{13}$$

and the width m of the neural network satisfies

$$m \geqslant \mathtt{C}'\left(\frac{n}{\lambda}\right)^3 (\ln m)^{\mathtt{C}} \ln(n/\varepsilon)^2, \tag{14}$$

for some large constants $\mathtt{C}, \mathtt{C}' > 0$. Then with high probability with respect to the random initialization, the training error decays exponentially,

$$\sum_{\beta=1}^n (f_\beta(t) - y_\beta)^2 \lesssim n e^{-\frac{\lambda t}{2n}},$$

which reaches ε at time $t \asymp (n/\lambda)\ln(n/\varepsilon)$.

The assumption that there exists $\lambda > 0$ such that $\lambda_{\min}\left[K_0^{(2)}(x_\alpha, x_\beta)\right]_{1\leqslant\alpha,\beta\leqslant n} \geqslant \lambda$ appears in many previous papers (Du et al., 2018b;a; Arora et al., 2019b). It is proven in (Du et al., 2018b;a), if no two inputs are parallel, the smallest eigenvalue of the kernel matrix is strictly positive. In general λ might depend on the size of the training data set n. For two layer neural networks with random training data, quantitative estimates for λ are obtained in (Ghorbani et al., 2019; Zhang et al., 2019; Xie et al., 2016). It is proven that with high probability with respect to the random training data, $\lambda \gtrsim n^\beta$ for some $0 < \beta < 1/2$.

It is proven in (Du et al., 2018a) that if there exists $\lambda^{(H)} > 0$,

$$\lambda_{\min}\left[G_0^{(H)}(x_\alpha, x_\beta)\right]_{1\leqslant\alpha,\beta\leqslant n} \geqslant \lambda^{(H)},$$

then for $m \geqslant \mathtt{C}(n/\lambda^{(H)})^4$ the gradient descent finds a global minimum. Corollary 13 improves this result in two ways: (i) We improve the quartic dependence of n to a cubic dependence. (ii) We recall that $K_t^{(2)} = \sum_{\ell=1}^{H+1} G_0^{(\ell)}$, and those kernels $G_0^{(\ell)}$ are all non-negative definite. The smallest eigenvalue of $K_0^{(2)}$ is typically much bigger than that of $G_0^{(H)}$, i.e., $\lambda \gg \lambda^{(H)}$. Moreover, since $K_t^{(2)}$ is a sum of $H+1$ non-negative definite operators, we expect that λ gets larger, if the depth H is larger.

The NTH, i.e., (9) and (10), is just an infinite sequence of relationship. It cannot be used to determine the dynamic of NTK. However, thanks to the a priori estimates of the higher order kernels (12), it holds that for any $p \leqslant p^*$ with high probability $\|K_t^{(p+1)}\|_\infty \lesssim (\ln m)^{\mathtt{C}}/m^{p/2}$. The derivative $\partial_t K_t^{(p)}$ is an expression involves the higher order kernel $K_t^{(p+1)}$, which is small provided that p is large enough. Therefore, we can approximate the original NTH (10) by simply setting $\partial_t K_t^{(p)} = 0$. In this way, we obtain the following truncated hierarchy of ordinary differential equations of p levels, which we call the truncated NTH,

$$\begin{aligned}
&\partial_t \tilde f_\alpha(t) = -\frac{1}{n}\sum_{\beta=1}^n \tilde K_t^{(2)}(x_\alpha, x_\beta)(\tilde f_\beta(t) - y_\beta)\\
&\partial_t \tilde K_t^{(r)}(x_{\alpha_1}, x_{\alpha_2}, \cdots, x_{\alpha_r})\\
&= -\frac{1}{n}\sum_{\beta=1}^n \tilde K_t^{(r+1)}(x_{\alpha_1}, x_{\alpha_2}, \cdots, x_{\alpha_r}, x_\beta)(\tilde f_\beta(t) - y_\beta),\\
&\partial_t \tilde K_t^{(p)}(x_{\alpha_1}, x_{\alpha_2}, \cdots, x_{\alpha_p}) = 0.
\end{aligned} \tag{15}$$

where $2 \leqslant r \leqslant p-1$, and

$$\begin{aligned}
&\tilde f_\beta(0) = f_\beta(0), \quad \beta = 1, 2, \cdots, n,\\
&\tilde K_0^{(r)} = K_0^{(r)}, \quad r = 2, 3, \cdots, p.
\end{aligned}$$

In the following theorem, we show this system of truncated equations (15) approximates the dynamic of the NTK up to arbitrary precision, provided that p is large enough.

Theorem 2.6. *Under Assumptions 2.1 and 2.2, we take an even $p \leqslant p^*$ and further assume that*

$$\lambda_{\min}\left[K_0^{(2)}(x_\alpha, x_\beta)\right]_{1\leqslant\alpha,\beta\leqslant n} \geqslant \lambda. \tag{16}$$

Then there exist constants $\mathtt{c}, \mathtt{C}, \mathtt{C}' > 0$ such that for t

$$t \leqslant \min\{\mathtt{c}\sqrt{\lambda m/n}/(\ln m)^{\mathtt{C}}, m^{\frac{p^*}{2(p^*+1)}}/(\ln m)^{\mathtt{C}'}\}, \tag{17}$$

the dynamic (9) can be approximated by the truncated dynamic (15),

$$\left(\sum_{\beta=1}^n (f_\beta(t) - \tilde f_\beta(t))^2\right)^{1/2} \lesssim \frac{(1+t)t^{p-1}\sqrt{n}}{m^{p/2}} \min\left\{t, \frac{n}{\lambda}\right\}, \tag{18}$$

and

$$\begin{aligned}
&|K_t^{(2)}(x_\alpha, x_\beta) - \tilde K_t^{(2)}(x_\alpha, x_\beta)|\\
&\lesssim \frac{(1+t)t^{p-1}}{m^{p/2}}\left(1 + \frac{(1+t)t(\ln m)^{\mathtt{C}}}{m}\min\left\{t, \frac{n}{\lambda}\right\}\right).
\end{aligned} \tag{19}$$

We remark that the error terms, i.e., the righthand sides of (18) and (19) can be arbitrarily small, provided that p is large enough. In other words, if we take p large enough, the truncated NTH (15) can approximate the original dynamic (9), (10) up to any precision provided that the time constraint (17) is satisfied. To better illustrate the scaling in Theorem 2.6, let us treat λ, ε, n as constants, ignore the $\log n, \log m$ factors and focus on the trade-off between time t and width m. Then (17) simplifies to $t \lesssim m^{p^*/2(p^*+1)}$. If p^* is sufficiently large, the exponent approaches $1/2$, and the condition is equivalent to that t is much smaller than $m^{1/2}$. In this regime, (18) simplifies to

$$\left(\sum_{\beta=1}^n (f_\beta(t) - \tilde f_\beta(t))^2\right)^{1/2} \lesssim \left(\frac{t}{\sqrt{m}}\right)^p, \tag{20}$$

and similarly (19) simplifies to

$$|K_t^{(2)}(x_\alpha, x_\beta) - \tilde K_t^{(2)}(x_\alpha, x_\beta)| \lesssim \left(\frac{t}{\sqrt{m}}\right)^p. \tag{21}$$

The error terms in (20) and (21) are smaller than any $\delta > 0$, provided we take $p \geqslant \log(1/\delta)/\log(\sqrt{m}/t)$.

Now if we take $t \asymp (n/\lambda)\ln(n/\varepsilon)$, so that Corollary 2.5 guarantees the convergence of the dynamics. Consider two special cases: (i) If we take $p = 2$, then the error in (18)

is $\mathrm{O}(n^{7/2}\ln(n/\varepsilon)^3/\lambda^3 m)$, which is negligible provided that the width m is much bigger than $n^{7/2}$. We conclude that if m is much bigger than $n^{7/2}$, the truncated NTH gives a complete description of the original dynamic of the NTK up to the equilibrium. The condition that m is much bigger than $n^{7/2}$ is better than the previous best available one which requires $m \gtrsim n^4$. (ii) If we take $p = 3$, then the error in (18) is $\mathrm{O}(n^{9/2}\ln(n/\varepsilon)^4/\lambda^4 m^{3/2})$ when $t \asymp (n/\lambda)\ln(n/\varepsilon)$, which is negligible provided that the width m is much bigger than n^3. We conclude that if m is much bigger than n^3, the truncated NTH gives a complete description of the original dynamic of the NTK up to the equilibrium. Finally, we note that the estimates in Theorem 2.6 clearly improved for smaller t.

The previous convergence theory of overparametrized neural networks works only for very wide neural networks, i.e., $m \gtrsim n^3$. For any width (not necessary that $m \gtrsim n^3$), Theorem 2.6 guarantees that the truncated NTH approximates the training dynamics of deep neural networks. The effect of the width appears in the approximation time and the error terms, (18) and (19), i.e., the wider the neural networks are, the truncated dynamic (15) approximates the training dynamic for longer time and the approximation error is smaller. We recall from (7) that the NTK is the sum of $H+1$ non-negative definite operators, $K_t^{(2)} = \sum_{\ell=1}^{H+1} G_t^{(\ell)}$. We expect that λ as in (16) gets bigger, if the depth H is larger. Therefore, large width and depth makes the truncated dynamic (15) a better approximation.

Thanks to Theorem 2.6, the truncated NTH (15) provides a good approximation for the evolution of the NTK. The truncated dynamic can be used to predict the output of new data points. Recall that the training data are $\{(x_\beta, y_\beta)\}_{1\leqslant\beta\leqslant n} \subset \mathbb{R}^d \times \mathbb{R}$. The goal is to predict the output of a new data point x. To do this, we can first use the truncated dynamic to solve for the approximated outputs $\{\tilde f_\beta(t)\}_{1\leqslant\beta\leqslant n}$. Then the prediction on the new test point $x \in \mathbb{R}^d$ can be estimated by sequentially solving the higher order kernels $\tilde K_t^{(p)}(x, \mathcal{X}^{p-1}), \tilde K_t^{(p-1)}(x, \mathcal{X}^{p-2}), \cdots, \tilde K_t^{(2)}(x, \mathcal{X})$ and $\tilde f_x(t)$,

$$
\begin{aligned}
&\partial_t \tilde f_x(t) = -\frac{1}{n}\sum_{\beta=1}^n \tilde K_t^{(2)}(x, x_\beta)(\tilde f_\beta(t) - y_\beta)\\
&\partial_t \tilde K_t^{(r)}(x, x_{\alpha_1}, x_{\alpha_2}, \cdots, x_{\alpha_{r-1}})\\
&= -\frac{1}{n}\sum_{\beta=1}^n \tilde K_t^{(r+1)}(x, x_{\alpha_1}, x_{\alpha_2}, \cdots, x_{\alpha_r-1}, x_\beta)(\tilde f_\beta(t) - y_\beta),\\
&\partial_t \tilde K_t^{(p)}(x, x_{\alpha_1}, x_{\alpha_2}, \cdots, x_{\alpha_{p-1}}) = 0.
\end{aligned}
\tag{22}
$$

where $2 \leqslant r \leqslant p-1$.

3. Technique overview

In general, the summands appearing in kernel $K_t^{(r)}(x_{\alpha_1}, x_{\alpha_2}, \cdots, x_{\alpha_r})$ are product of inner products of vectors obtained in the following way: starting from one of the vectors

$$
\frac{a_t}{\sqrt{m}}, \quad \frac{\mathbf{1}}{\sqrt{m}}, \quad \{x_\beta^{(1)}, x_\beta^{(2)}, \cdots, x_\beta^{(H)}\}_{\beta\in\{\alpha_1,\alpha_2,\cdots,\alpha_r\}},
$$

(i) multiply one of the matrices

$$
\begin{aligned}
&\left\{\frac{W_t^{(2)}}{\sqrt{m}}, \frac{(W_t^{(2)})^\top}{\sqrt{m}}, \cdots, \frac{W_t^{(H)}}{\sqrt{m}}, \frac{(W_t^{(H)})^\top}{\sqrt{m}}\right\},\\
&\{\sigma_1'(x_\beta), \sigma_2'(x_\beta), \cdots, \sigma_H'(x_\beta)\}_{\beta\in\{\alpha_1,\alpha_2,\cdots,\alpha_r\}};
\end{aligned}
\tag{23}
$$

(ii) multiply one of the matrices

$$
\mathrm{diag}(\cdots), \quad \sigma^{(s)}(x_\beta)\underbrace{\mathrm{diag}(\cdots)\cdots\mathrm{diag}(\cdots)}_{s-1 \text{ terms}},
\tag{24}
$$

for $s \geqslant 2$, where $\mathrm{diag}(\cdots)$ is the diagonalization of a vector obtained by recursively using (i) and (ii).

To describe the vectors appearing in $K_t^{(r)}(x_{\alpha_1}, x_{\alpha_2}, \cdots, x_{\alpha_r})$ in a formal way, we need to introduce some more notations. We denote $\mathfrak{D}_0$ the set of expressions in the following form

$$
\mathfrak{D}_0 := \{\mathsf{e}_s\mathsf{e}_{s-1}\cdots\mathsf{e}_1\mathsf{e}_0 : 0 \leqslant s \leqslant 4H-3\}
\tag{25}
$$

where e_j is chosen from the following sets:

$$
\mathsf{e}_0 \in \left\{a_t, \{\sqrt{m}x_\beta^{(1)}, \sqrt{m}x_\beta^{(2)}, \cdots, \sqrt{m}x_\beta^{(H)}\}_{1\leqslant\beta\leqslant n}\right\}
$$

and for $1 \leqslant j \leqslant s$,

$$
\begin{aligned}
\mathsf{e}_j \in &\left\{\left\{\frac{W_t^{(2)}}{\sqrt{m}}, \frac{(W_t^{(2)})^\top}{\sqrt{m}}, \cdots, \frac{W_t^{(H)}}{\sqrt{m}}, \frac{(W_t^{(H)})^\top}{\sqrt{m}}\right\},\right.\\
&\left.\{\sigma_1'(x_\beta), \sigma_2'(x_\beta), \cdots, \sigma_H'(x_\beta)\}_{1\leqslant\beta\leqslant n}\right\}.
\end{aligned}
$$

We remark that from expression (7), each summand in $K_t^{(2)}(x_{\alpha_1}, x_{\alpha_2})$ is of the form

$$
\frac{\langle \mathsf{v}_1(t), \mathsf{v}_2(t)\rangle}{m}, \quad \frac{\langle \mathsf{v}_1(t), \mathsf{v}_2(t)\rangle}{m}\frac{\langle \mathsf{v}_3(t), \mathsf{v}_4(t)\rangle}{m},
$$

where $\mathsf{v}_1(t), \mathsf{v}_2(t), \mathsf{v}_3(t), \mathsf{v}_4(t) \in \mathfrak{D}_0$. But the set $\mathfrak{D}_0$ contains more terms than those appearing in $K_t^{(2)}(\cdot,\cdot)$. Given that we have constructed $\mathfrak{D}_0, \mathfrak{D}_1, \cdots, \mathfrak{D}_r$, we denote $\mathfrak{D}_{r+1}$ the set of expressions in the following form

$$
\mathfrak{D}_{r+1} := \{\mathsf{e}_s\mathsf{e}_{s-1}\cdots\mathsf{e}_1\mathsf{e}_0 : 0 \leqslant s \leqslant 4H-3\},
\tag{26}
$$

where e_j is chosen from the following sets (notice that we have included $\mathbf{1}$ in the following set, which does not appear in the definition of $\mathfrak{D}_0$):

$$\mathsf{e}_0 \in \left\{a_t, \mathbf{1}, \{\sqrt{m}x_\beta^{(1)}, \sqrt{m}x_\beta^{(2)}, \cdots, \sqrt{m}x_\beta^{(H)}\}_{1\leqslant\beta\leqslant n}\right\},$$

and for $1 \leqslant j \leqslant s$, e_j belongs to one of the sets

$$\begin{aligned}
&\left\{\left\{\frac{W_t^{(2)}}{\sqrt{m}}, \frac{(W_t^{(2)})^\top}{\sqrt{m}}, \cdots, \frac{W_t^{(H)}}{\sqrt{m}}, \frac{(W_t^{(H)})^\top}{\sqrt{m}}\right\},\right.\\
&\quad \left.\{\sigma_1'(x_\beta), \sigma_2'(x_\beta), \cdots, \sigma_H'(x_\beta)\}_{1\leqslant\beta\leqslant n}\right\},\\
&\{\mathrm{diag}(\mathsf{d}), \quad \mathsf{d} \in \mathfrak{D}_0 \cup \mathfrak{D}_1 \cup \cdots \cup \mathfrak{D}_r\},\\
&\left\{\sigma_\ell^{(u+1)}(x_\beta)\mathrm{diag}(\mathsf{d}_1)\mathrm{diag}(\mathsf{d}_2)\cdots\mathrm{diag}(\mathsf{d}_u) : 1 \leqslant \ell \leqslant H,\right.\\
&\left.1 \leqslant \beta \leqslant n, 1 \leqslant u \leqslant r, \mathsf{d}_1, \mathsf{d}_2, \cdots, \mathsf{d}_u \in \mathfrak{D}_0 \cup \mathfrak{D}_1 \cup \cdots \cup \mathfrak{D}_r\right\}.
\end{aligned}$$

Moreover, the total number of diag operations in the expression $\mathsf{e}_s\mathsf{e}_{s-1}\cdots\mathsf{e}_1\mathsf{e}_0 \in \mathfrak{D}_{r+1}$ is exactly $r+1$. We remark that if $\mathsf{d} \in \mathfrak{D}_s$, then it contains s diag operations. On the other hand, by definition, we view $\mathrm{diag}(\mathsf{d})$ as an element with $s+1$ diag operations because the diag in $\mathrm{diag}(\mathsf{d})$ counted as one diag operation.

We will show in the supplementary materials that each summand in $K_t^{(r)}(x_{\alpha_1}, x_{\alpha_2}, \cdots, x_{\alpha_r})$ is of the form

$$\begin{aligned}
&\frac{1}{m^{r/2-1}}\prod_{j=1}^{s}\frac{\langle \mathsf{v}_{2j-1}(t), \mathsf{v}_{2j}(t)\rangle}{m}, \quad 1 \leqslant s \leqslant r,\\
&\mathsf{v}_1(t), \mathsf{v}_2(t), \cdots, \mathsf{v}_{2s}(t) \in \mathfrak{D}_0 \cup \mathfrak{D}_1 \cup \cdots \cup \mathfrak{D}_{r-2}.
\end{aligned} \tag{27}$$

The initial value $K_0^{(r)}(x_{\alpha_1}, x_{\alpha_2}, \cdots, x_{\alpha_r})$ can be estimated by successively conditioning based on the depth of the neural network. A convenient scheme is given by the tensor program (Yang, 2019), which was developed to characterize the scaling limit of neural network computations. In the supplementary materials, we show at time $t = 0$, those vectors $\mathsf{v}_j(0)$ in (27) are combinations of projections of independent Gaussian vectors. As a consequence, we have that $\langle \mathsf{v}_{2j-1}(0), \mathsf{v}_{2j}(0)\rangle/m$ concentrates around certain constant with high probability. So does the product $\prod_{j=1}^{s}\langle \mathsf{v}_{2j-1}(0), \mathsf{v}_{2j}(0)\rangle/m$. This gives the claim (11).

In the supplementary materials, we consider the quantity:

$$\xi(t) = \max\{\|\mathsf{v}_j(t)\|_\infty : \mathsf{v}_j(t) \in \mathfrak{D}_0 \cup \mathfrak{D}_1 \cup \cdots \cup \mathfrak{D}_{p^*-1}\}.$$

Again using the tensor program, we show that with high probability $\|\mathsf{v}_j(0)\|_\infty \lesssim (\ln m)^{\mathsf{c}}$. This gives the estimate of $\xi(t)$ at $t = 0$. Next we show that the (p^*+1)-th derivative of $\xi(t)$ can be controlled by itself. This gives a self-consistent differential equation of $\xi(t)$:

$$\partial_t^{(p^*+1)}\xi(t) \lesssim \frac{\xi(t)^{2p^*}}{m^{p^*/2}}. \tag{28}$$

Combining with the initial estimate of $\xi(t)$, it follows that for time $0 \leqslant t \leqslant m^{\frac{p^*}{2(p^*+1)}}/(\ln m)^{\mathsf{c}'}$, it holds that $\xi(t) \lesssim (\ln m)^{\mathsf{c}}$. Especially $\|\mathsf{v}_j(t)\|_\infty \lesssim (\ln m)^{\mathsf{c}}$. Then the claim (12) in Theorem 2.3 follows.

Thanks to the a priori estimate (12), we show that along the continuous time gradient descent, the higher order kernels $K_t^{(r)}$ vary slowly. We prove Corollary 2.4 and 2.5, and Theorem 2.6 in the supplementary materials by a Grönwall type argument.

4. Discussion and future directions

In this paper, we study the continuous time gradient descent (gradient flow) of deep fully-connected neural networks. We show that the training dynamic is given by a data dependent infinite hierarchy of ordinary differential equations, i.e., the NTH. We also show that this dynamic of the NTH can be approximated by a finite truncated dynamic up to any precision. This description makes it possible to directly study the change of the NTK for deep neural networks. Here we list some future directions.

1. We mainly study deep fully-connected neural networks in this paper, we believe the same statements can be proven for convolutional and residual neural networks.

2. We focus on the continuous time gradient descent for simplicity. Our approach developed here can be generalized to analyze discrete time gradient descent with small step size. We elaborate the main idea here. The discrete time gradient descent is given by

$$\theta_{t+1} = \theta_t - \eta\nabla_\theta L(\theta_t) = \theta_t - \frac{\eta}{n}\sum_{\beta=1}^{n}\nabla_\theta f_\beta(t)(f_\beta(t) - y_\beta),$$

where η is the learning rate. We write the NTK as $\mathcal{K}^{(2)}(x_\alpha, x_\beta; \theta_t)$ to make the dependence on θ_t explicit. To estimate the NTK $\mathcal{K}^{(2)}(x_{\alpha_1}, x_{\alpha_2}; \theta_{t+1})$ at time $t+1$, we use the taylor expansion,

$$\begin{aligned}
\mathcal{K}^{(2)}(x_{\alpha_1}, x_{\alpha_2}; \theta_{t+1}) &= \mathcal{K}^{(2)}(x_{\alpha_1}, x_{\alpha_2}; \theta_t) + \sum_{r=3}^{s-1}\frac{(-\eta)^r}{n^r}\\
&\sum_{1\leqslant\beta_1,\beta_2,\cdots,\beta_{r-2}\leqslant n}\mathcal{K}^{(r)}(x_{\alpha_1}, x_{\alpha_2}, x_{\beta_1}, \cdots, x_{\beta_{r-2}}; \theta_t)\\
&\times (f_{\beta_1}(t) - y_{\beta_1})\cdots(f_{\beta_{r-2}}(t) - y_{\beta_{r-2}}) + \Omega_s,
\end{aligned} \tag{29}$$

where Ω_s is the error term in the truncation and the higher order kernels $\mathcal{K}^{(r)}$ are given by

$$\begin{aligned}
&\mathcal{K}^{(r)}(x_{\alpha_1}, x_{\alpha_2}, x_{\beta_1}, \cdots, x_{\beta_{r-2}}; \theta_t)\\
&= \nabla_\theta^{(r-2)}\mathcal{K}^{(2)}(x_{\alpha_1}, x_{\alpha_2}; \theta_t)(\nabla_\theta f_{\beta_1}(t), \cdots, \nabla_\theta f_{\beta_{r-2}}(t)).
\end{aligned}$$

We notice that $\mathcal{K}^{(r)}$ is different from $K^{(r)}$ in the NTH hierarchy. A similar argument as for (12) can be used to derive

the a priori estimates of these kernels $\mathcal{K}^{(r)}$. We expect to have that $\|\mathcal{K}^{(r)}\|_\infty \lesssim (\ln m)^{\mathsf{C}}/m^{r/2-1}$ with high probability with respect to the random initialization. Therefore $\|\Omega_s\|_\infty \leqslant \mathrm{O}((\ln m)^{\mathsf{C}}\eta^s/n^2m^{s/2-1})$, which can be arbitrarily small provided that $\eta \ll \sqrt{m}$ and s is large enough. Similar procedure can be applied to NTH in the discrete time dynamics and we will not get into details here. To conclude this discussion, we believe that, under the assumption $\eta \ll \sqrt{m}$, our analysis in continuous time can be carried over to the discrete time dynamics.

3. It will be interesting to further analyze the behaviors of the truncated dynamics (15), and understand why it is better than kernel regression using the limiting NTK. For example, if we truncate the dynamic at $p = 3$,

$$\begin{aligned} &\partial_t \tilde{f}_\alpha(t) = -\frac{1}{n}\sum_j \tilde{K}_t^{(2)}(x_\alpha, x_\beta)(\tilde{f}_\beta(t) - y_\beta), \\ &\partial_t \tilde{K}_t^{(2)}(x_{\alpha_1}, x_{\alpha_2}) \\ &= -\frac{1}{n}\sum_\beta \tilde{K}_t^{(3)}(x_{\alpha_1}, x_{\alpha_2}, x_\beta)(\tilde{f}_\beta(t) - y_\beta), \\ &\partial_t \tilde{K}_t^{(3)}(x_{\alpha_1}, x_{\alpha_2}, x_{\alpha_3}) = 0. \end{aligned} \tag{30}$$

The difference between (30) and the kernel regression using the limiting NTK is that in (30), the kernel $K_t^{(2)}$ changes along time at a rate of $\mathrm{O}((\ln m)^{\mathsf{C}}/m)$. We denote the residue vector as $\tilde{r}(t) = (\tilde{f}_1(t) - y_1, \tilde{f}_2(t) - y_2, \cdots, \tilde{f}_n(t) - y_n)^\top$. Since $\tilde{K}_t^{(3)}$ does not depend on time t, we can integrate the second equation in (30) to get $K_t^{(2)} = K_t^{(0)} - \frac{1}{n}K_0^{(3)}[\int_0^t \tilde{r}(s)\mathrm{d}s]$. We denote $\tilde{R}(t) = \int_0^t \tilde{r}(s)\mathrm{d}s$, and plug the previous relation to the first equation of (30), to obtain the following system of ordinary differential equations

$$\begin{aligned} &\partial_t \tilde{r}(t) = -\frac{1}{n}K_0^{(2)}[\tilde{r}(t)] - \frac{1}{n}\tilde{K}_0^{(3)}[\tilde{R}(t), \tilde{r}(t)], \\ &\partial_t \tilde{R}(t) = \tilde{r}(t). \end{aligned}$$

The above system of ordinary differential equations can be easily solved numerically. It will be interesting to understand, under what conditions, the change of the kernel $K_t^{(2)}$ helps optimization and generalization.

4. The optimal condition to use the NTH approximation. We have shown that $m \gtrsim n^3$ guarantees the approximation of the NTH to the deep neural network up to both of them find global minimizers. Our analysis loses a factor n by estimating the norm of the kernels $K^{(r)}$using their entry-wise L_∞ norm. This results in a loss of a factor $1/n$. We believe that this loss can be recovered by a more careful analysis and one can improve the condition to $m \gtrsim n^2$. To further improve on this condition, one will have to analyze other cancellation effects. Assuming such an analysis is possible, one might reach the condition $m \gtrsim n$. It would be interesting to see if this is the best possible condition for approximating deep neural networks by the NTH, i.e., if one can show with some example that deep neural network converges for $m \ll n$, while the approximation by the NTH fails.

Acknowledgements

The work of J.H. is supported by the Institute for Advanced Study. The work of H.-T. Y. is partially supported by NSF Grants DMS-1606305 and DMS-1855509, and a Simons Investigator award. The authors would like to thank the reviewers for their thoughtful comments and efforts towards improving our manuscript.

References

Allen-Zhu, Z. and Li, Y. What can resnet learn efficiently, going beyond kernels? *arXiv preprint arXiv:1905.10337*, 2019.

Allen-Zhu, Z., Li, Y., and Liang, Y. Learning and generalization in overparameterized neural networks, going beyond two layers. *arXiv preprint arXiv:1811.04918*, 2018a.

Allen-Zhu, Z., Li, Y., and Song, Z. A convergence theory for deep learning via over-parameterization. *In ICML, arXiv:1811.03962*, 2018b.

Araújo, D., Oliveira, R. I., and Yukimura, D. A mean-field limit for certain deep neural networks. *arXiv preprint arXiv:1906.00193*, 2019.

Arora, S., Du, S. S., Hu, W., Li, Z., Salakhutdinov, R., and Wang, R. On exact computation with an infinitely wide neural net. *arXiv preprint arXiv:1904.11955*, 2019a.

Arora, S., Du, S. S., Hu, W., Li, Z., and Wang, R. Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks. *arXiv preprint arXiv:1901.08584*, 2019b.

Bhojanapalli, S., Neyshabur, B., and Srebro, N. Global optimality of local search for low rank matrix recovery. In *Advances in Neural Information Processing Systems*, pp. 3873–3881, 2016.

Cao, Y. and Gu, Q. Generalization bounds of stochastic gradient descent for wide and deep neural networks. In *Advances in Neural Information Processing Systems*, pp. 10835–10845, 2019.

Chizat, L. and Bach, F. On the global convergence of gradient descent for over-parameterized models using optimal transport. In *Advances in neural information processing systems*, pp. 3036–3046, 2018.

Choromanska, A., Henaff, M., Mathieu, M., Ben Arous, G., and LeCun, Y. The loss surfaces of multilayer networks. In *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics*, pp. 192–204, 2015.

Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., and Kuksa, P. Natural language processing (almost) from scratch. *Journal of machine learning research*, 12(Aug):2493–2537, 2011.

Dauphin, Y. N., Pascanu, R., Gulcehre, C., Cho, K., Ganguli, S., and Bengio, Y. Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. In *Advances in Neural Information Processing Systems*, pp. 2933–2941, 2014.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Du, S. S., Lee, J. D., Li, H., Wang, L., and Zhai, X. Gradient descent finds global minima of deep neural networks. *ICML, arXiv:1811.03804*, 2018a.

Du, S. S., Zhai, X., Poczos, B., and Singh, A. Gradient descent provably optimizes over-parameterized neural networks. *In ICLR, arXiv:1810.02054*, 2018b.

Dyer, E. and Gur-Ari, G. Asymptotics of wide networks from feynman diagrams. *arXiv preprint arXiv:1909.11304*, 2019.

Ge, R., Huang, F., Jin, C., and Yuan, Y. Escaping from saddle points—online stochastic gradient for tensor decomposition. In *Proceedings of The 28th Conference on Learning Theory*, pp. 797–842, 2015.

Ge, R., Lee, J. D., and Ma, T. Matrix completion has no spurious local minimum. In *Advances in Neural Information Processing Systems*, pp. 2973–2981, 2016.

Ghorbani, B., Mei, S., Misiakiewicz, T., and Montanari, A. Linearized two-layers neural networks in high dimension. *arXiv preprint arXiv:1904.12191*, 2019.

Glorot, X. and Bengio, Y. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 249–256, 2010.

Hinton, G., Deng, L., Yu, D., Dahl, G., Mohamed, A.-r., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Kingsbury, B., et al. Deep neural networks for acoustic modeling in speech recognition. *IEEE Signal processing magazine*, 29, 2012.

Jacot, A., Gabriel, F., and Hongler, C. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in neural information processing systems*, pp. 8571–8580, 2018.

Jin, C., Ge, R., Netrapalli, P., Kakade, S. M., and Jordan, M. I. How to escape saddle points efficiently. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 1724–1732. JMLR. org, 2017.

Kawaguchi, K. Deep learning without poor local minima. In *Advances in Neural Information Processing Systems*, pp. 586–594, 2016.

Kawaguchi, K. and Huang, J. Gradient descent finds global minima for generalizable deep neural networks of practical sizes. *arXiv preprint arXiv:1908.02419*, 2019.

Kawaguchi, K. and Kaelbling, L. P. Elimination of all bad local minima in deep learning. *arXiv preprint arXiv:1901.00279*, 2019.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pp. 1097–1105, 2012.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Lee, J., Xiao, L., Schoenholz, S. S., Bahri, Y., Sohl-Dickstein, J., and Pennington, J. Wide neural networks of any depth evolve as linear models under gradient descent. *arXiv preprint arXiv:1902.06720*, 2019.

Lee, J. D., Simchowitz, M., Jordan, M. I., and Recht, B. Gradient descent only converges to minimizers. In *Conference on learning theory*, pp. 1246–1257, 2016.

Li, Y. and Liang, Y. Learning overparameterized neural networks via stochastic gradient descent on structured data. In *Advances in Neural Information Processing Systems*, pp. 8157–8166, 2018.

Liang, S., Sun, R., Lee, J. D., and Srikant, R. Adding one neuron can eliminate all bad local minima. In *Advances in Neural Information Processing Systems*, 2018.

Mei, S., Misiakiewicz, T., and Montanari, A. Mean-field theory of two-layers neural networks: dimension-free bounds and kernel limit. *arXiv preprint arXiv:1902.06015*, 2019.

Nguyen, P.-M. Mean field limit of the learning dynamics of multilayer neural networks. *arXiv preprint arXiv:1902.02880*, 2019.

Park, D., Kyrillidis, A., Caramanis, C., and Sanghavi, S. Non-square matrix sensing without spurious local minima via the burer-monteiro approach. *arXiv preprint arXiv:1609.03240*, 2016.

Sainath, T. N., Mohamed, A.-r., Kingsbury, B., and Ramabhadran, B. Deep convolutional neural networks for lvcsr. In *2013 IEEE international conference on acoustics, speech and signal processing*, pp. 8614–8618. IEEE, 2013.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., et al. Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016.

Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., et al. Mastering the game of go without human knowledge. *Nature*, 550(7676):354, 2017.

Sirignano, J. and Spiliopoulos, K. Mean field analysis of deep neural networks. *arXiv preprint arXiv:1903.04440*, 2019.

Song, M., Montanari, A., and Nguyen, P. A mean field view of the landscape of two-layers neural networks. *Proceedings of the National Academy of Sciences*, 115:E7665–E7671, 2018.

Song, Z. and Yang, X. Quadratic suffices for over-parametrization via matrix chernoff bound. *arXiv preprint arXiv:1906.03593*, 2019.

Sun, J., Qu, Q., and Wright, J. Complete dictionary recovery over the sphere i: Overview and the geometric picture. *IEEE Transactions on Information Theory*, 63(2):853–884, 2016.

Sun, J., Qu, Q., and Wright, J. A geometric analysis of phase retrieval. *Foundations of Computational Mathematics*, 18 (5):1131–1198, 2018.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. Going deeper with convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–9, 2015.

Wu, Y., Schuster, M., Chen, Z., Le, Q. V., Norouzi, M., Macherey, W., Krikun, M., Cao, Y., Gao, Q., Macherey, K., et al. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*, 2016.

Xie, B., Liang, Y., and Song, L. Diverse neural network learns true target functions. *arXiv preprint arXiv:1611.03131*, 2016.

Yang, G. Scaling limits of wide neural networks with weight sharing: Gaussian process behavior, gradient independence, and neural tangent kernel derivation. *CoRR*, abs/1902.04760, 2019. URL http://arxiv.org/abs/1902.04760.

Yehudai, G. and Shamir, O. On the power and limitations of random features for understanding neural networks. *arXiv preprint arXiv:1904.00687*, 2019.

Zhang, G., Martens, J., and Grosse, R. B. Fast convergence of natural gradient descent for over-parameterized neural networks. In *Advances in Neural Information Processing Systems*, pp. 8080–8091, 2019.

Zou, D. and Gu, Q. An improved analysis of training over-parameterized deep neural networks. In *Advances in Neural Information Processing Systems*, pp. 2053–2062, 2019.

Zou, D., Cao, Y., Zhou, D., and Gu, Q. Stochastic gradient descent optimizes over-parameterized deep relu networks. *arXiv preprint arXiv:1811.08888*, 2018.