# On Random Kernels of Residual Architectures

**Etai Littwin**[1] **Tomer Galanti**[1] **Lior Wolf**[1]

[1]The Blavatnik School of Computer Science, Tel Aviv University, Tel Aviv, Israel

## Abstract

We analyze the finite corrections to the neural tangent kernel (NTK) of residual and densely connected networks, as a function of both depth and width. Surprisingly, our analysis reveals that given a fixed depth, residual networks provide the best tradeoff between the parameter complexity and the coefficient of variation (normalized variance), followed by densely connected networks and vanilla MLPs. While in networks that do not use skip connections, convergence to the NTK requires one to fix the depth, while increasing the layers' width. Our findings show that in ResNets, convergence to the NTK may occur when depth and width simultaneously tend to infinity, provided with a proper initialization. In DenseNets, however, the convergence of the NTK to its limit as the width tends to infinity is guaranteed, at a rate that is independent of both the depth and scale of the weights. Our experiments validate the theoretical results and demonstrate the advantage of deep ResNets and DenseNets for kernel regression with random gradient features.

## 1 INTRODUCTION

Understanding the effect of different architectures on the ability to train deep networks has long been a major research topic. A popular playing ground for studying the forward and backward propagation of signals at the point of initialization, is the "infinite width" regime [Neal, 1996, Sirignano and Spiliopoulos, 2019, Schoenholz et al., 2017, Yang and Schoenholz, 2017, Lee et al., 2018, Arora et al., 2019]. In this regime, Gaussian Process behaviour emerges in pre-activations, when the weights are sampled i.i.d. from a normal distribution, giving rise to tractable training dynamics [Jacot et al., 2018, Lee et al., 2019, 2018, de G. Matthews et al., 2018].

This notion was first made precise by the Neural Tangent Kernel (NTK) paper [Jacot et al., 2018], in which it is shown that the training dynamics of fully connected networks trained with gradient descent can be characterized by a kernel when the width of the network approaches infinity. Specifically, the evolution through time of the function computed by the network follows the dynamics of kernel regression. Let $f(x; w) \in \mathbb{R}$ denote the output of a fully connected feedforward network of width $n$, with i.i.d. normally distributed weights $w$ and input $x \in \mathbb{R}^{n_0}$. The *neural tangent kernel* (NTK) is given by:

$$\mathcal{K}(x, x'; w) := \langle \frac{\partial f(x; w)}{\partial w}, \frac{\partial f(x'; w)}{\partial w} \rangle \tag{1}$$

with:

$$\mathring{\mathcal{K}}(x, x') := \mathbb{E}_w[\mathcal{K}(x, x'; w)], \tag{2}$$

where $w$ is the vector that concatenates the weights in $f(x; w)$. As shown in [Jacot et al., 2018], when the width tends to infinity, minimizing the squared loss $\mathcal{L}(w)$ using gradient descent is equivalent to a kernel regression with kernel $\mathring{\mathcal{K}}$.

Recent empirical support has demonstrated the power of NTK and specifically CNTK (convolutional neural tangent kernel) on practical datasets, showing new state-of-the-art results for kernel methods, surpassing other known kernels by a wide margin [Arora et al., 2019, Yu et al., 2020, Arora et al., 2020].

In [Geiger et al., 2019], the variance of the empirical NTK was directly connected with the generalization performance of a trained neural network. When the width of the network exceeds a threshold $n^\star$, it is shown that the generalization gap is affected by both the fluctuations of the NTK at initialization as a function of the randomness in the weight and its evolution during training. Further, it is shown that the former source dominates the latter. In other words, the distance between the empirical NTK and its expected value

*Proceedings of the Thirty-Seventh Conference on Uncertainty in Artificial Intelligence* (UAI 2021), PMLR 161:897–907.

is the main source of randomness affecting post-training performance.

Therefore, in this work, we analyze the variance of the NTK at initialization. To compare different architectures, we use the coefficient of variation (normalized variance) of the NTK, which is a unitless measure of variation, to quantify the variance of the NTK. Namely,

$$\mathrm{V}(\mathcal{K}(x,x';w)) := \frac{Var\big(\mathcal{K}(x,x';w)\big)}{\mathbb{E}[\mathcal{K}(x,x';w)]^2} \tag{3}$$

This quantity has recently been addressed in the case of vanilla feedforward fully connected networks [Hanin and Nica, 2020], where it was shown that the normalized variance of the diagonal entries of the NTK is exponential in the ratio between the depth $L$ and width $n$:

$$\mathrm{V}(\mathcal{K}(x,x;w)) \approx \exp\left[\frac{CL}{n}\right] - 1, \tag{4}$$

where $C > 0$ is a constant. Hence, convergence to the asymptotic kernel cannot happen when both are taken to infinity at the same rate. From Eq. 4 it is evident that for an $L$-depth vanilla network, the width should be at least $\Omega(L)$ in order to maintain a fixed ratio in the exponent of Eq. 4. In this case, the total parameter complexity of the network is at least $\Omega(L^3)$. This important observation suggests that deep and narrow vanilla networks operate far from the "infinite width" regime at initialization. In this work, we derive finite width and depth corrections to the NTK of residual and densely connected architectures, revealing a depth invariant property unique to these architectures. From this analysis, it is evident that, in contrast to vanilla ReLU networks, the required parameter complexities of $L$-depth ResNets and DenseNets is as small as $\mathcal{O}(L)$ and $\mathcal{O}(L^2)$ (resp.) to maintain a bounded normalized variance.

Our theoretical analysis considers only the diagonal entries of the NTK, which from Eq. 1, correspond to output gradients squared norm. As such, it is able to capture the pathological implications on back-propagation in these architectures by revealing how gradient norms concentrate at large depths. Since only individual entries along the diagonal are investigated theoretically, we complement the analysis with an empirical investigation of the joint distribution of the full NTK matrix. We conduct extensive empirical experiments on MNIST and multiple small UCI datasets using random draws of $\mathcal{K}$ as kernel approximations, demonstrating the power of random gradient features $\nabla_w f(x;w)$ of deep residual architectures. Surprisingly, for fixed-width ResNets and DenseNets, the performance of kernel regression using $\mathcal{K}$ as a substitute for $\mathring{\mathcal{K}}$ improve with depth and approach the latter, whereas, in vanilla architectures, clear degradation is observed.

Our main contributions are as follows.

1. Thms. 1 and 2 introduce a forward-backward norm propagation duality for a wide family of ReLU feedforward architectures, which is a useful tool for analyzing the rate of convergence of $\mathcal{K}(x,x;w)$, for finite sized networks.
2. In Thms. 3 and 4, we rigorously derive finite width and depth corrections for ResNet and DenseNet architectures, revealing a fundamentally different relationship between width, depth and $\mathcal{K}(x,x;w)$. Unlike vanilla architectures, when properly scaled, convergence to the asymptotic kernel is achieved, when taking both the width and the depth of the architecture to infinity simultaneously.
3. Our experiments validate the convergence rates of both the diagonal $\mathcal{K}(x,x;w)$ and off-diagonal $\mathcal{K}(x,x';w)$ NTK terms. Furthermore, they demonstrate the advantage of deep ResNets and DenseNets over vanilla networks for kernel regression with random gradient features on MNIST and multiple small UCI datasets.

## 2 RELATED WORK

The study of wide neural networks has been at the forefront of theoretical deep learning research in the last few years. A number of papers [Yu et al., 2020, Lee et al., 2019, Arora et al., 2019] have followed up on the original NTK work [Jacot et al., 2018]. An extension of the GP and NTK results is given in [Yang, 2019], where it is shown that neural networks of any architecture (including weight-tied ResNets, DenseNets, or RNNs) converge to GPs in the infinite width limit, and prove the existence of the infinite width NTKs. In [Lee et al., 2019], corrections to the NTK are derived to bound the change of the NTK during training, which applies for both the diagonal and off-diagonal entries of the NTK. However, depth is treated as a constant, and therefore their result only apply for shallow networks. An interesting problem is to quantify the convergence rate of the NTK to its limit. Feynman diagrams were used to provide finite width corrections to the NTK [Dyer and Gur-Ari, 2020]. However, the analysis relies on a conjecture and does not hold for residual and densely connected architectures. What is most related to our results are the finite width corrections to the NTK for vanilla networks, introduced in [Hanin and Nica, 2020]. These results depend on the depth of the network. However, their analysis does not apply to residual architectures. In contrast, in our Thms. 1 and 2, we establish a duality that exists between forward and backward statistics, which allows considering only forward statistics and can be readily applied for most fully connected architectures, with arbitrary topologies. In [Hanin and Rolnick, 2018] they tackle two failure modes that are caused in finite-size networks by exponential explosion or decay of the norm of intermediate layers. It is shown that for random fully connected vanilla ReLU networks, the variance of the squared norm of the activations exponentially increases, even when initializing with the $\frac{2}{fan-in}$ initialization [He et al., 2015]. For ResNets, this failure mode can be overcome by correctly rescaling the residual branches. However, it is not clear how such a rescaling affects the backpropagation of gradients.

# 3 GENERALIZATION AS A FUNCTION OF VARIANCE

Recent work has shown that the generalization of trained neural networks follows a double descent curve as a function of width. Counter-intuitively, when the width $n$ exceeds some threshold $n^\star$, the test error decreases until it reaches a local minimum at $n = \infty$. In [Geiger et al., 2019], a framework was proposed to describe how generalization depends on the width of the network through analysis of initial fluctuations in function space at initialization, as well as how randomness propagates through the course of training. In this section, we briefly summarize their analysis.

Let $f_T(x)$ denote the output of a network after $T$ iterations of gradient descent, and let $\bar{f}_T(x) = \mathbb{E}_w[f_T(x)]$ denote the expected value of $f_T(x)$ over different initializations. Then we have, $f_T(x) = \bar{f}_T(x) + \delta f_T(x)$, where $\delta f_T$ arises from the random draw of the weights. In regression tasks, we can describe how the fluctuations $\delta f_T(x)$ affect generalization using the following simple identity:

$$\begin{aligned}\Delta\epsilon :=& \mathbb{E}_w[\mathbb{E}_x\|\bar{f}_T(x) + \delta f_T(x) - y(x)\|^2] \\ &- \mathbb{E}_w[\mathbb{E}_x\|\bar{f}_T(x) - y(x)\|^2] = \mathbb{E}_{w,x}[\|\delta f_T(x)\|^2],\end{aligned} \tag{5}$$

where $x$ is distributed according to some population distribution $D$, $y$ is the target function that we would like to learn, and $\Delta\epsilon$ is the contribution to the generalization error due to $\delta f_T$. This identity follows from the fact that for any $x$,

$$\begin{aligned}&\mathbb{E}_w[\langle \overline{f}_T(x) - y(x), \delta f_T(x)\rangle] \\ =&\langle \overline{f}_T(x) - y(x), \mathbb{E}_w[\delta f_T(x)]\rangle = 0,\end{aligned} \tag{6}$$

since $\overline{f}_T(x) - y(x)$ is independent of $w$.

To understand how $\delta f_T$ behaves as $n$ increases, we must understand how the initial randomness of the weights propagates through the course of training. Note that the variance of the initial function $f_0(x)$ does not vanish as $n \to \infty$, as the output function approaches a GP in that limit. We note that the term $\delta f_T$ can be decomposed in the following manner:

$$\delta f_T(x) = f_0(x) - \bar{f}_T(x) + \sum_{t=1}^{T} \Delta f_t(x), \tag{7}$$

where under gradient descent with a learning rate $\mu$[1], $\Delta f_t(x)$ is approximately given by the following dynamics:

$$\begin{aligned}\Delta f_t(x) =& -\mu \sum_i \mathring{\mathcal{K}}(x, x_i) \cdot \frac{\partial \mathcal{L}(f_t(x))}{\partial f(x)} \\ &- \mu \sum_i \left(\mathcal{K}_t(x, x_i) - \mathring{\mathcal{K}}(x, x_i)\right) \cdot \frac{\partial \mathcal{L}(f_t(x))}{\partial f(x)}\end{aligned} \tag{8}$$

[1]Technically Eq. 8 holds approximately up to a $\mathcal{O}(1/n)$ correction.

The variance of the first term is bounded when $f_t(x)$ is bounded through training since $\mathring{\mathcal{K}}(x, x_i)$ has a zero variance. On the other hand, in general, the second term tends to be unbounded. In order to upper bound the second term, we can use the triangle inequality,

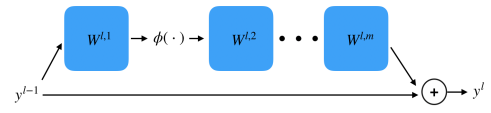
$$\begin{aligned}\|\mathcal{K}_t(x, x_i) - \mathring{\mathcal{K}}(x, x_i)\| \leq& \|\mathcal{K}_t(x, x_i) - \mathcal{K}_0(x, x_i)\| \\ &+ \|\mathcal{K}_0(x, x_i) - \mathring{\mathcal{K}}(x, x_i)\|\end{aligned}$$

We have, therefore, decoupled the change in the NTK to a term that depends on how the NTK changes from its initialized value during training, and the difference between its initial value and its expected value. Note that both terms are random due to the randomness of the weights. As indicated in [Lee et al., 2019, Dyer and Gur-Ari, 2020, Littwin et al., 2020a, Aitken and Gur-Ari, 2020], the first term scales as $\mathcal{O}(1/n)$, while the second term scales as $\mathcal{O}(1/n^{0.5})$ [Geiger et al., 2019]. In a few recent papers, minimizing the fluctuations of the second term while keeping some computational envelope constant is shown to improve generalization [Littwin et al., 2020b, Geiger et al., 2019, Golubeva et al., 2020]. Motivated by these findings, in this paper, we analyze the fluctuations of the second term as a function of both depth and width.

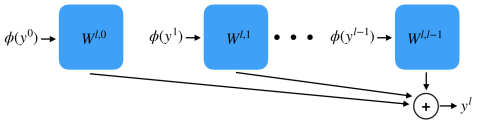
# 4 PRELIMINARIES AND NOTATIONS

Throughout the paper, we make use of the following notations. Let $f(x; w) \in \mathbb{R}$ denote the output of a parameterized function $f$ on input $x \in \mathbb{R}^{n_0}$ with a vector $w$ of real valued parameters. We assume that the coordinates of $w$ are normally distributed i.i.d samples. With no loss of generality, we also assume that $\|x\|_2 = 1$. Throughout the paper we use the ReLU non-linearity, $\phi(x) := \max(0, x)$. Our analysis can be readily extended to Leaky ReLU as well. The intermediate outputs of a neural network are denoted by $\{y^l(x)\}_{l=0}^L$ (see Eqs. 9 and 10), for a fixed input $x \in \mathbb{R}^{n_0}$. For simplicity, the dependence of the outputs on $x$ is often made implicit $\{y^l\}_{l=0}^L$ when the specific input used to calculate the outputs can be inferred from the context. $y_i^l$ denotes the $i$'th component of the vector $y^l$, and $n_1, ..., n_L$ denote the width of the corresponding layers, with $n_0$ being the input dimension. We denote by $\|x\|_2$ the Euclidean norm of the vector $x$ and by $\|W\|_2$ the Frobenius norm of the matrix $W$. We denote the weight matrix associated with layer $l$ by $W^l$, with lower case letters $w_{i,j}^l$ denoting the individual components of $W^l$. Additional superscripts $W^{l,k}$ are used, when several weight matrices are associated with layer $l$. Weights, $w$, appearing without any superscript denote all the weights concatenated into a vector. The NTK of the function $f$ is denoted by $\mathcal{K}(x, x'; w) := \frac{\partial f(x;w)}{\partial w} \cdot \frac{\partial^\top f(x';w)}{\partial w}$. We denote by $f = \mathcal{O}(g)$ the big-O notation, $f = \Omega(g)$ when $g = \mathcal{O}(f)$ and $f = \Theta(g)$ if $f = \mathcal{O}(g)$ and $f = \Omega(g)$.

**Residual networks** have reintroduced the concept of bypass connections [He et al., 2016], allowing the training of deep



(a)



(b)

Figure 1: Illustrating **(a)** ResNet and **(b)** DenseNet, as in Eq. 9 and 10 (with constant width and absent scaling coefficients).

and narrow models with relative ease. A generic, residual architecture $f(x;w)$, with residual branches of depth $m$, takes the form: $f(x;w) = \frac{1}{\sqrt{n_L}} \cdot W^L \cdot y^L$, where, for all $l \in [L]$, $y^l$ is defined recursively as follows:

$$y^l = \begin{cases} \frac{1}{\sqrt{n_0}} \cdot W^0 x & l = 0 \\ y^{l-1} + \sqrt{\alpha_l} y^{l-1,m} & o.w \end{cases}$$
$$y^{l-1,h} = \begin{cases} \sqrt{\frac{1}{n_{l-1,h-1}}} W^{l,h} q^{l-1,h-1} & 1 < h \leq m \\ \sqrt{\frac{1}{n_{l-1}}} \cdot W^{l,h} y^{l-1} & h = 1 \end{cases} \quad (9)$$

Here, $\{\alpha_l\}_{l=1}^L$ are scaling coefficients, $W^0 \in \mathbb{R}^{n_0' \times n_0}$, $W^{l,h} \in \mathbb{R}^{n_{l-1,h} \times n_{l-1,h-1}}$, $W^{l,1} \in \mathbb{R}^{n_{l-1,1} \times n_{l-1}}$, $W^{l,m} \in \mathbb{R}^{n_l \times n_{l-1,m-1}}$, $q^{l,h} = \sqrt{2}\phi(y^{l,h})$ (see Fig. 1 for an illustration).

**DenseNets** were recently introduced [Huang et al., 2017], demonstrating faster training, as well as improved performance on several popular datasets. The main architectural features introduced by DenseNets include the connection of each layer output to all subsequent layers, using concatenation operations, instead of summation, such that the weights of layer $l$ multiply the concatenation of the outputs $y^0, ..., y^{l-1}$. A DenseNet $f(x;w)$ is defined in the following manner: $f(x;w) := \frac{1}{\sqrt{n_L}} \cdot W^L \cdot y^L$, where, for all $l \in [L]$, $y^l$ is defined recursively as follows:

$$y^l = \begin{cases} \frac{1}{\sqrt{n_0}} \cdot W^0 x & l = 0 \\ \sqrt{\frac{\alpha}{n_{l-1} \cdot l}} \sum_{h=0}^{l-1} W^{l,h} q^h & o.w \end{cases} \quad (10)$$

where $\alpha$ is a scaling coefficient, $q^l = \sqrt{2}\phi(y^l)$, $W^{l,h} \in \mathbb{R}^{n_l \times n_{l-1}}$ (see Fig. 1 for an illustration).

# 5 FORWARD-BACKWARD NORM PROPAGATION DUALITY

In this work, we aim to derive an expression for the first and second moments of the diagonal entries $\mathcal{K}(x,x;w)$ at the point of initialization $w$, given by the Jacobian squared norm evaluated on $x$:

$$\mathcal{K}(x,x;w) = \|J(x)\|_2^2 = \sum_{\mathbf{k}} \|J^{\mathbf{k}}(x)\|_2^2, \quad (11)$$

where $J^{\mathbf{k}}(x) := \frac{\partial f}{\partial W^{\mathbf{k}}}$ denotes the per-weight Jacobian. Bold letters (a.k.a $\mathbf{k}, \mathbf{u}, \mathbf{v}$) stand for identities of matrices in the network. For instance, in ResNets, $\mathbf{k}$ can take values in $\{0, L\} \cup [L] \times [m]$. The sum $\sum_{\mathbf{k}} \|J^{\mathbf{k}}\|_2^2$ denotes summation over every weight matrix in the network. In the following analysis, we assume that the output of $f$ is computed using a single fixed sample $x$. To facilitate our derivation, we introduce a link between the propagation of the norm of the activations and the norm of the per-layer Jacobian in random ReLU networks of finite width and depth. This link will then allow us to study the statistical properties of the full Jacobian in general architectures incorporating residual connections and concatenations with relative ease. Specifically, we would like to establish a connection between the first and second moments of the squared norm of the output $f(x;w)^2$, and those of the per layer Jacobian norm $\|J^{\mathbf{k}}\|_2^2$. Using a path-based notation, for any weight matrix $W^{\mathbf{k}}$, the output $f(x;w)$ can be decomposed to paths that go through $W^{\mathbf{k}}$ (i.e, paths that include weights from $W^{\mathbf{k}}$), denoted by $f_{\mathbf{k}}(x;w)$, and paths that skip $W^{\mathbf{k}}$, denoted by the complement $f_{\mathbf{k}}^c(x;w)$. Namely:

$$\begin{aligned} f(x;w) &= f_{\mathbf{k}}(x;w) + f_{\mathbf{k}}^c(x;w) \\ &= \sum_{\gamma \in S_{\mathbf{k}}} c_\gamma z_\gamma \prod_{l=1}^{|\gamma|} w_{\gamma,l} + \sum_{\gamma \in S \setminus S_{\mathbf{k}}} c_\gamma z_\gamma \prod_{l=1}^{|\gamma|} w_{\gamma,l}, \end{aligned} \quad (12)$$

where the summations are over paths $\gamma \in S$ from input to output, with $|\gamma|$ denoting the length of the path, and $c_\gamma$ a scaling factor. In standard fully connected networks, we have $|\gamma| = L+2$ (when considering the initial and final projections $W^0, W^L$) and the total number of paths is $\prod_{l=0}^L n_l$. The term $z_\gamma \prod_{l=1}^{|\gamma|} w_{\gamma,l}$ denotes the product of weights along path $\gamma$, multiplied by a binary variable $z_\gamma \in \{0,1\}$, indicating whether path $\gamma$ is active (i.e all relevant activations along the specific path are on). The set $S_{\mathbf{k}}$ indicates the set of all paths that include weights from $W^{\mathbf{k}}$.

We make the following definition:

**Definition 1 (Reduced network)** *Let $f(x;w)$ be a neural network (e.g., vanilla network, ResNet or DenseNet). We define the reduced network $f_{(\mathbf{k})}(x;w)$ to be the neural network obtained by removing all connections bypassing weights $W^{\mathbf{k}}$ from the network $f(x;w)$. The corresponding hidden*

*layers of $f_{(\mathbf{k})}(x;w)$ are denoted by $y^0_{(\mathbf{k})},...,y^L_{(\mathbf{k})}$ and its weights by $w_{(\mathbf{k})}$.*

Note that for vanilla networks, it holds that, for all $\mathbf{k} \in [L]$, we have: $f_{(\mathbf{k})}(x;w) = f_{\mathbf{k}}(x;w) = f(x;w)$ and $y^l_{(\mathbf{k})} = y^l$. In the general case, the equality $f_{(\mathbf{k})}(x;w) = f_{\mathbf{k}}(x;w)$ does not hold, since $f_{(\mathbf{k})}(x;w)$ contains different activation patterns, induced by the removal of residual connections. The following theorem states that surprisingly, the moments of both are equal in the family of considered ReLU networks (see Fig. 2 for an illustration):

**Theorem 1** *Let $f(x;w)$ be a ResNet/DenseNet. Then, for any non-negative even integer $m$, we have:*

$$\forall\, \mathbf{k}:\ \mathbb{E}_w\left[(f_{(\mathbf{k})}(x;w))^m\right] = \mathbb{E}_w\left[(f_{\mathbf{k}}(x;w))^m\right] \tag{13}$$

The following theorem relates the moments of $\|J^{\mathbf{k}}\|_2^2$ with those of $f_{(\mathbf{k})}(x;w)$:

**Theorem 2** *Let $f(x;w)$ be a ResNet/DenseNet. Then, we have for all $k$:*

1. $\mathbb{E}_w\left[\|J^{\mathbf{k}}\|_2^2\right] = \mathbb{E}_w\left[(f_{(\mathbf{k})}(x;w))^2\right]$.
2. $\frac{\mathbb{E}_w\left[(f_{(\mathbf{k})}(x;w))^4\right]}{3} \leq \mathbb{E}_w\left[\|J^{\mathbf{k}}\|_2^4\right] \leq \mathbb{E}_w\left[(f_{(\mathbf{k})}(x;w))^4\right]$.

From Eq. 11 and Thm. 2, we can derive bounds on the second moment of $\mathcal{K}(x,x;w)$, by observing the moments of $f_{(\mathbf{k})}(x;w)$. In addition, Thm. 2 also allows us to derive bounds on the convergence rate of $\mathcal{K}(x,x;w)$ to $\mathbb{E}_w[\mathcal{K}(x,x;w)] = \mathring{\mathcal{K}}(x,x)$, given by the ratio:

$$\eta(n,L) := \frac{\mathbb{E}_w[\mathcal{K}(x,x;w)^2]}{\mathbb{E}_w[\mathcal{K}(x,x;w)]^2} = \frac{Var\big(\mathcal{K}(x,x;w)\big)}{\mathbb{E}_w\left[\mathcal{K}(x,x;w)\right]^2} + 1 \tag{14}$$

In general, the tools developed in Thms. 1-2 can be used for analyzing a wide range of feedforward network architectures. Specifically, in Thms. 3 and 4, we derive bounds on the asymptotic behavior of $\eta$ for ResNet and DenseNet architectures, with respect to both width and depth.

**Theorem 3** *Let $f(x;w)$ be a depth $L$, constant width ResNet with residual branches of depth $m$ (Eq. 10 with $n'_0, n_l, n_{l,h} = n$ for all $l \in [L]$ and $h \in [m]$), with positive initialization constants $\{\alpha_l\}_{l=1}^L$. Then, there exists a constant $C > 0$ such that:*

$$\max\left[1, \frac{\sum_{\mathbf{u}} \alpha^2_{l_u}}{\sum_{\mathbf{u},\mathbf{v}} \alpha_{l_u}\alpha_{l_v}} \cdot \xi\right] \leq \eta(n,L) \leq \xi \tag{15}$$

*where:*

$$\xi = \exp\left[\frac{5m}{n} + \frac{C}{n}\sum_{l=1}^{L}\frac{\alpha_l}{1+\alpha_l}\right] \cdot (1 + \mathcal{O}(1/n)) \tag{16}$$

First, for clarity, we note that we always have $\frac{\sum_{\mathbf{u}} \alpha^2_{l_u}}{\sum_{\mathbf{u},\mathbf{v}} \alpha_{l_u}\alpha_{l_v}} \leq 1$ as the sum in the denominator includes the sum in the numerator (and each summand is non-negative). For example, by choosing $\alpha_l = \alpha > 0$ for all $l$, the ratio is $1/L$. From the result of Thm. 3, it is evident that the convergence rate is exponential in $\frac{m}{n} + \frac{1}{n}\sum_{l=1}^{L}\alpha_l$. This result supports the selection of a small $m$, as reflected in the common practice to have a small depth for the residual branches, since $\xi$ scales exponentially with respect to $m/n$. In addition, when setting $\{\alpha_l\}_{l=1}^L$, such that, $\frac{1}{n}\sum_{l=1}^{L}\alpha_l$ vanishes as $n$ tends to infinity, ensures the convergence of $\eta$ to 1, regardless of depth. Note that by selecting $\{\alpha_l\}_{l=1}^L$, such that, $\sum_{l=1}^{L}\alpha_l \approx \mathcal{O}(1)$ is sufficient (although not necessary), and was also suggested in [Zhang et al., 2019] as a way to train ResNets without batchnorm [Ioffe and Szegedy, 2015]. Our results, however, reveal a much stronger implication of this initialization, as it also bounds the fluctuations of the squared Jacobian norm, implying a closer relationship with the "kernel regime" at the initialization of deep ResNets. From Thm. 3, we conclude that a proper initialization plays a crucial role in determining the asymptotic behavior of $\eta$ in deep ResNets. Surprisingly, this relationship between initialization and $\eta$ breaks down when considering DenseNets, as illustrated in the following theorem.

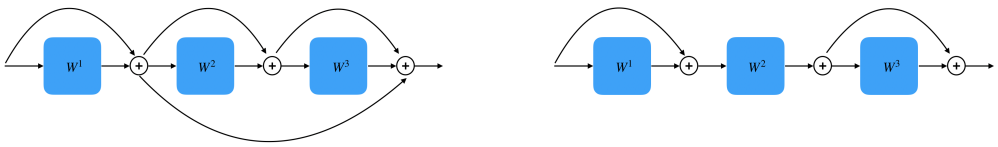
**Theorem 4** *Let $f(x;w)$ be a constant width DenseNet (Eq. 10 with $n'_0, n_l = n$ for all $l \in [L]$), with initialization constant $\alpha > 0$. Then, there exist constants $C_1, C_2 > 0$, such that:*

$$\max\left[1, \frac{C_1}{L\log(L)^2} \cdot \xi\right] \leq \eta(n,L) \leq \xi \tag{17}$$

*where:*

$$\xi = \exp\left[C_2/n\right] \cdot (1 + \mathcal{O}(1/n)) \tag{18}$$

Surprisingly, the depth parameter $L$, as well as the initialization scale $\alpha$ are absent in the upper bound of Eq. 17, revealing a depth and scale-invariant property unique to DenseNets. In other words, the convergence rate of $\eta$ to 1 is exponential in $\frac{C_2}{n}$, and does not depend on depth, or the scaling coefficient of the weights. This property represents a fundamental unique aspect of DenseNets, which might explain the practical advantages observed in models incorporating dense residual connections. It is important to stress that it is impossible to replicate the guarantees presented in Thms. 3 and 4 by simply normalizing the network differently. That is because, the expression $\eta(n,L)$ is invariant to the scale of the weights, i.e., its value does not change when multiplying $f(x;w)$ by a constant. Therefore, maintaining a bounded normalized variance of the NTK of a $L$-depth network comes at the cost of a different parameter complexity for each architecture. This is formulated in the following remark.




(a) (b)

Figure 2: **An illustration of Thm. 1.** The activations of the network in **(a)** are completely different from those of the network in **(b)**, in which all skip connections bypassing layer $l = 2$ are removed. However, the moments of the gradient norms at layer $l = 2$ are exactly the same in both **(a)** and **(b)**.

**Remark 1** *For DenseNets and ResNets (with $\alpha_l = 1/L$ and $m = 2$), it is possible to choose a constant width $n = \mathcal{O}(1)$ (independent of L) while maintaining a bounded NTK variance. In this case, the overall number of parameters in DenseNets is $\mathcal{O}(L^2)$. On the other hand, in ResNets, the overall number of parameters is $\mathcal{O}(L)$, as each one of its $L$ layers contributes a constant number of parameters $2n^2 = \mathcal{O}(1)$. However, in vanilla models, it is required that the width $n$ grow linearly with depth in order to maintain a bounded variance. Therefore, each layer contributes $\Omega(L^2)$ parameters, and the overall number of parameters is $\Omega(L^3)$. The added efficiency is the product of an inherent architectural advantage brought forth by the ResNet architecture.*

**Off-diagonal entries in deep linear networks** Our analysis in this work holds for the diagonal entries of the NTK, and the extension to the off-diagonal entries is not straightforward in the general case. However, extending our analysis to the off-diagonal entries is immediate in the case of deep linear networks. Indeed for these models, similar results can be easily extended for any NTK entry by combining Prop. 6 in the appendix with the proofs of Thms. 3 and 4.

## 6 EXPERIMENTS

To validate our theoretical observations, we conducted a series of experiments using the MNIST, CIFAR10, and 43 small UCI datasets (see Tab. 1 in the appendix for the list). Throughout the experiments, the report the average and standard deviation of the accuracy rate at test time over 20 runs. Our default initialization values are $\alpha := \alpha_1 = ... = \alpha_L = 0.1/L$ for ResNets and $\alpha = 0.5$ for DenseNets.

### 6.1 NORMALIZED VARIANCE OF NTK

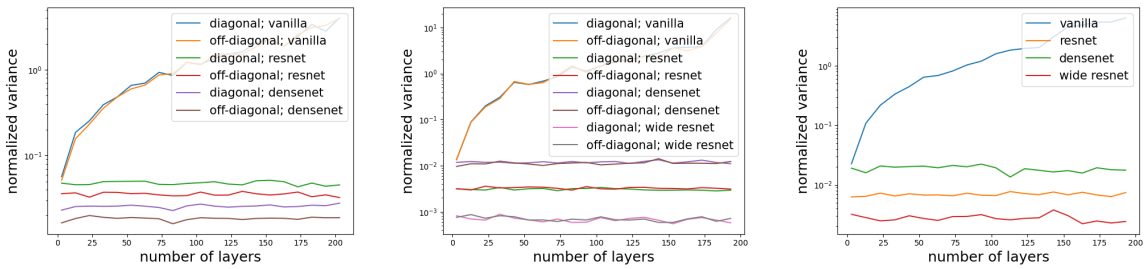
To validate our theory, we conducted an experiment for estimating the coefficient of variation of the NTK. For each model, we fixed the width to be $n = 500$, varied the number of layers and for each depth, we estimated the value in Eq. 3 for $x = x'$ and for $x \neq x'$. In order to estimate these terms, we sampled 5000 different vectors $w$ for $f(x; w)$ from a standard normal distribution and estimated $\mathrm{V}(\mathcal{K}(x, x; w))$ and $\mathrm{V}(\mathcal{K}(x, x'; w))$ empirically. In Fig. 3(a-b), we use synthetic samples $x$ and $x'$, that are generated as follows: $x = \hat{x}/\|\hat{x}\|_2$ and $x_2 = \hat{x}'/\|\hat{x}'\|_2$ are two vectors, such that, each coordinate of $\hat{x}$ is distributed according to $\mathcal{N}(0.5, 1)$ and each coordinate of $\hat{x}'$ is distributed according to $\mathcal{N}(-0.5, 1)$. In Fig. 3(c), we used the CIFAR10 data samples.

In Fig. 3(a-b) we plot the normalized variance of the diagonal and off-diagonal elements of the kernel as a function of the number of layers for the various architectures. In Fig. 3(c) we plot the normalized variance of the full-kernel matrix over 1000 samples, i.e., we estimate $\frac{\mathbb{E}_w[\|\mathcal{K}_w - \mathbb{E}_w[\mathcal{K}_w]\|_2^2}{\|\mathbb{E}_w[\mathcal{K}_w]\|_2^2}$, where $\mathcal{K}_w = (\mathcal{K}(x_i, x_j; w))_{i,j}$ is the empirical kernel matrix. In Fig. 3(a), we compare the behaviours of fully connected architectures, and in Figs. 3(b-c), we compare between convolutional architectures, including the wide ResNet architecture [Zagoruyko and Komodakis, 2016]. The results are plotted in log-scale. As can be seen, the normalized variance of the diagonal and off-diagonal elements of the kernel are highly correlated for all architectures. In addition, for residual and dense architectures, the normalized variance of the NTK is relatively constant when varying the number of layers, while for vanilla networks, the normalized variance of the NTK grows exponentially.

We conducted a similar experiment to study the effect of the depth of each block in ResNets on the normalized variance of the NTK. The setting is the same as in Fig. 3, except that we vary the depth of each residual block instead of the number of blocks, which stays constant (4 or 10). In Fig. 4(a), we plot the results for fully-connected ResNets on synthetic samples and in Fig. 4(b) for convolutional ResNets on samples from CIFAR10. As can be seen, the diagonal and off-diagonal elements of the NTK scale exponentially with the depth of each block, as predicted in Thm. 3.

### 6.2 REGRESSION WITH RANDOM GRADIENT FEATURES

We conducted various experiments to compare the ability of the gradients $\nabla_w f(x; w)$ of each architecture to serve as random features for kernel regression. As we show, the fluc-




(a) (b) (c)

Figure 3: **Normalized variance of NTK for various models.** The x-axis stands for the number of layers. In **(a-b)** the y-axis stands for the values of $\mathrm{V}(\mathcal{K}(x,x';w))$ in log-scale. The diagonal terms specify the value for $x = x'$ and the off-diagonal terms specify the value for $x \neq x'$. In **(c)** the y-axis stands for the values of the deviations of the full-kernel matrix in log-scale. **(a)** Results for fully connected networks and **(b-c)** results for convolutional networks.

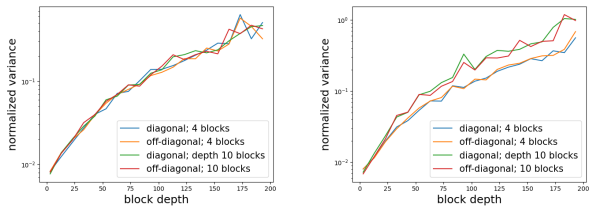



Figure 4: **Normalized variance of NTK for ResNets with 4 and 10 residual blocks.** The x-axis stands for the depth of each block. The y-axis stands for the values of $\mathrm{V}(\mathcal{K}(x,x';w))$ in log-scale. The diagonal terms specify the value for $x = x'$ and the off-diagonal terms specify the value for $x \neq x'$. **(left)** results for fully-connected ResNets and **(right)** results for convolutional ResNets.

tuations are highly correlated and have a dramatic effect on the performance of kernel regression. The process is as follows: for a given network $f(x;w)$, we sampled $w_1,\ldots,w_T$ at random from a standard normal distribution and used $\nabla_{w_i} f(x;w_i)$ as our random features. In addition, the labels are being cast into one-hot vectors corresponding to their discrete values in $[k]$. To solve the kernel regression task, we employed the closed form solution:

$$g(x;w) := (\mathcal{K}_T(x,x_1),\ldots,\mathcal{K}_T(x,x_m)) \cdot H_T^{-1} \cdot Y \quad (19)$$

where $\mathcal{K}_T(x,x') = \frac{1}{T}\sum_{i=1}^{T}\mathcal{K}(x,x';w_i)$, $H_k = (\mathcal{K}_T(x_i,x_j))_{i,j\in[m]} \in \mathbb{R}^{m\times m}$ and $Y \in \mathbb{R}^{m\times k}$ is a matrix whose $i$'th row is $y_i$.

**Experiments on MNIST** In this set of experiments, each training run was done over 2000 randomly selected MNIST training samples, where each train/test sample is normalized to have norm 1. In Fig. 5(a-c) we report the expected accuracy rates of $g(x;w)$ on the test set, when varying the number of layers of a fully connected network $f(x;w)$, while fixing the width to be $n \in \{50,100,500\}$ and $T = 1$. The performances of the infinite-width limit kernels of vanilla networks, ResNets, and DenseNets are plotted as well, under the names, 'vanilla kernel', 'resnet kernel' and 'densenet kernel' respectively. Their error bars are taken with respect to the selection of the training samples. In Fig. 5(d-f) we report the same results, when the width is $n \in \{2,50,100\}$ and $T = 30$. As can be seen, when fixing the width of the network, increasing the depth of a vanilla network is adverse to the performance of the kernel regression. However, this is not the case with ResNets and DenseNets. In addition, the results of performing kernel regression with the NTKs are comparable to the results of their corresponding infinite-width limit kernels.

In Fig. 6(a-c), we report the effect of varying the width when fixing the depth and $T = 1$. As can be seen, the performance of a standard network is significantly inferior to the performances of the kernel regressions corresponding to ResNets and DenseNets when the number of layers is larger than 4. It is evident that the performance of each architecture improves when increasing the width, however, standard neural networks are required to be much wider, in order to achieve the same degree of success as ResNets and DenseNets and convergence to the performance of the asymptotic kernel (which is denoted by 'ker; depth = $n$').

**Experiments on UCI** We also compared the performance of kernel regression over 43 small UCI datasets (see list in Tab. 1 in the appendix). We note that the performance of the various methods vary from one dataset to another as a result of dataset complexity, number of classes, etc. Therefore, in order to average the results over the various datasets, instead of reporting the absolute accuracy rates, we report the relative accuracy rates with respect to the accuracy rate of a three-layered network (i.e., the accuracy rate divided by the accuracy rate obtained with three layers). For each fully connected architecture, we compared the relative accuracy rates for widths 10, 100, 500, when varying the number of layers. The relative accuracy rates are averaged over the 43 datasets. The results, presented in Fig. 6(d-f), show that the performance of kernel regression for ResNet

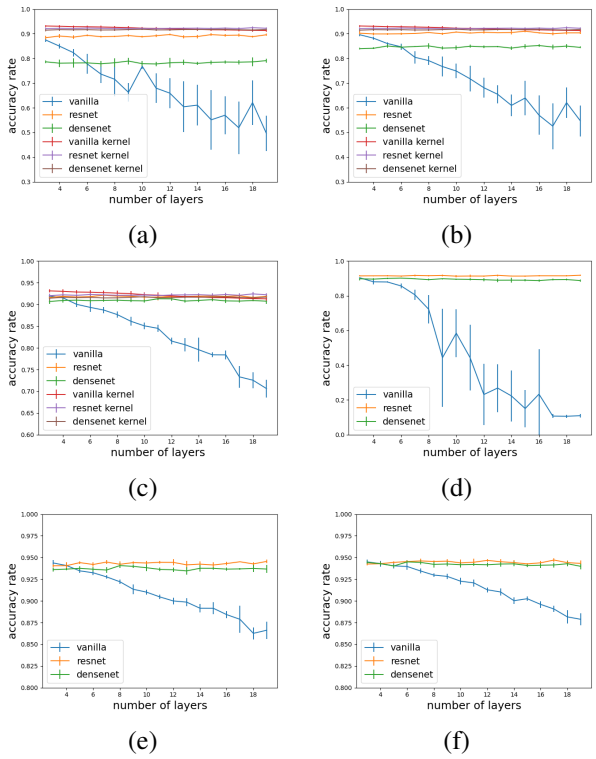



Figure 5: **Results on MNIST for kernel regression over random gradient features.** We plot the averaged accuracy rates, when varying the number of layers. In (a-b) $T = 1$ and the width of $f(x; w)$ is either **(a)** 50 **(b)** 100 or **(c)** 500. 'vanilla kernel', 'resnet kernel' and 'densenet kernel' stand for the results of the infinite-width limit kernels of vanilla networks, ResNets and DenseNets (resp.). In (d-f) $T = 30$ and the width of $f(x; w)$ is either **(d)** 2 **(e)** 10 or **(f)** 100.

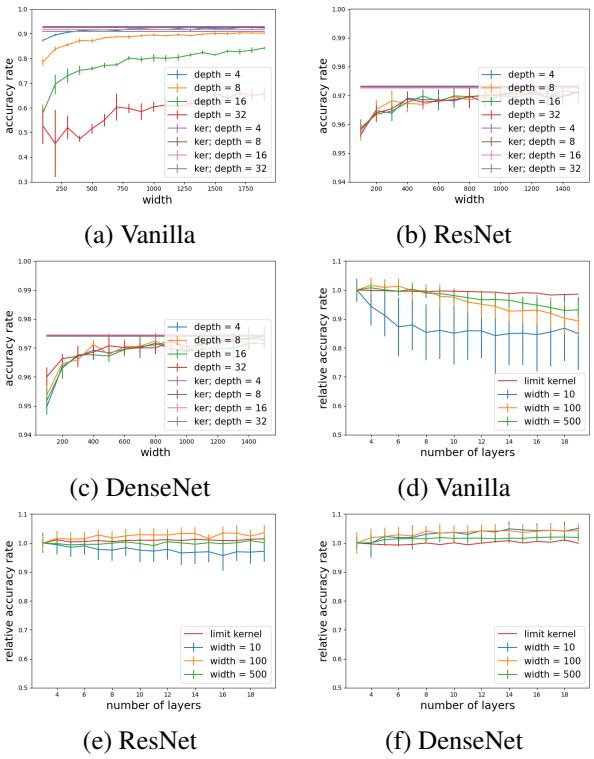
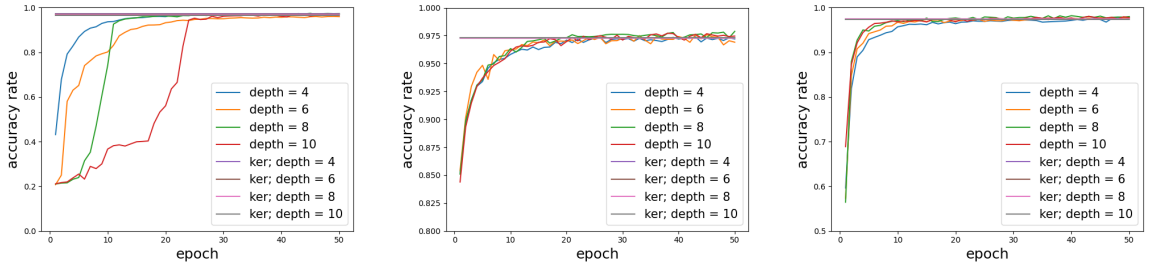



Figure 6: **(a-c) Results of kernel regression over random gradient features on MNIST.** Plotted are the averaged accuracy rates, when fixing $T = 1$ and varying the width of the three architectures: vanilla networks, ResNets, and DenseNets. 'ker; depth = $n$' stands for the results of the infinite-width limit kernel for depth $n$. **(d-f) Results on UCI.** Plotted are the averaged relative accuracy rates, when varying the depth for the three architectures. 'limit kernel' stands for the results of the infinite-width limit kernel.

and DenseNet architectures does not degrade as a result of increasing the number of layers. In fact, the results improve when increasing the number of layers for DenseNets and DenseNets of widths 100 (about 4-5% improvement). In contrast, for vanilla networks, where increasing the number of layers harms the performance. It is evident that when increasing the width of the vanilla network, the kernel regression performance becomes more stable but still degrades when increasing the number of layers. For comparison, we also plot the results of the infinite-width limit kernels under the name 'limit kernel'. Since Fig. 6(d-f) does not compare the performance of the various architectures, rather it compares its stability, for completeness, in Tab. 1 in the appendix we report the absolute accuracy rates of the various architectures with three layers and widths 10, 100, and 500. As can be seen, the different models achieve very similar results on all datasets.

**Experiments on CIFAR10 with CNTK** We compared the performance of Convolutional NTKs (CNTK) regressors on the CIFAR10 dataset. In these experiments $f(x; w)$ is a convolutional vanilla, residual or densely connected neural network. Each convolutional layer consists of $3 \times 3$ convolutions with stride 1 and padding 1. We treat the number of channels as the width of a given architecture.

Each training run was done over 4000 randomly selected CIFAR10 samples. In Fig. 8(a-c) we report the performance of the ResNet and DenseNet regressors when varying the number of layers and fixing the width of the various models. As can be seen, the performance of the ResNet and DenseNet regressors do not degrade as a function of depth, in contrast to vanilla networks. In Fig. 8(d-f), we report the performance of the various regressors when varying the width and fixing the depth to be $3, 6$ or $12$ (resp.). As evident from the plots, the performance of each architecture improves when increasing the width, however, vanilla networks are required to be much wider, in order to achieve the same degree of success as ResNets and DenseNets and convergence to the performance of the asymptotic kernel.




(a) Vanilla (b) ResNet (c) DenseNet

Figure 7: **SGD training for the three models of varying depths.** The x-axis specifies the epoch and the y-axis specifies the accuracy rate. **(a)** Results of vanilla MLPs, **(b)** Results of ResNets and **(c)** Results of DenseNets. The optimization stability of ResNets and DenseNets is unaffected by the depth, in contrast to vanilla networks.

### 6.3 SGD EXPERIMENTS

We compared the performance of the various architectures with varying depths on the MNIST dataset. Each model was trained for $50$ epochs, using SGD with a learning rate $\mu = 0.01$ and batch size $100$. For each model, we compare the performance of depths $2, 4, 6, 8$ and the performance of kernel regression with the corresponding width-limit kernels. As can be seen in Fig. 7, the performance of vanilla MLPs at the first epochs is worse for higher depths, and therefore, the overall optimization is slower. In contrast, for ResNets with a properly conditioned $\alpha = 0.1/L$, the performance is similar for different depths throughout the optimization. Furthermore, for DenseNets, it is evident that the performance is stable for different depths, even when the initialization parameter $\alpha = 0.5$ is independent of depth.

## 7 CONCLUSIONS

The Neural Tangent Kernel has provided new insights into the training dynamics of wide neural networks. In this work, we have derived finite width and depth corrections for ResNet and DenseNet architectures, and have shown convergence properties of deep residual models that are absent in the vanilla fully connected architectures. Our results shed new light on the effect of residual connections on the training dynamics of practically sized networks, suggesting that models incorporating residual connections operate much closer to the "kernel regime" approximation than vanilla architectures, even at large depths.

### Author Contributions

Etai Littwin and Tomer Galanti contributed equally to this paper.

### Acknowledgements

This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant ERC CoG 725974).

### References

Kyle Aitken and Guy Gur-Ari. On the asymptotics of wide networks with polynomial activations, 2020.

Sanjeev Arora, Simon S. Du, Wei Hu, Zhiyuan Li, Ruslan Salakhutdinov, and Ruosong Wang. On exact computation with an infinitely wide neural net. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NeurIPS, 2019.

Sanjeev Arora, Simon S. Du, Zhiyuan Li, Ruslan Salakhutdinov, Ruosong Wang, and Dingli Yu. Harnessing the power of infinitely wide deep nets on small-data tasks. In *International Conference on Learning Representations*, 2020. URL `https://openreview.net/forum?id=rkl8sJBYvH`.

Alexander G. de G. Matthews, Jiri Hron, Mark Rowland, Richard E. Turner, and Zoubin Ghahramani. Gaussian process behaviour in wide deep neural networks. In *International Conference on Learning Representations*, ICLR, 2018. URL `https://openreview.net/forum?id=H1-nGgWC-`.

Ethan Dyer and Guy Gur-Ari. Asymptotics of wide networks from feynman diagrams. In *International Conference on Learning Representations*, ICLR, 2020.

M. Geiger, Arthur Jacot, S. Spigler, F. Gabriel, Levent Sagun, Stéphane d'Ascoli, G. Biroli, C. Hongler, and M. Wyart. Scaling description of generalization with number of parameters in deep learning. *ArXiv*, abs/1901.01608, 2019.

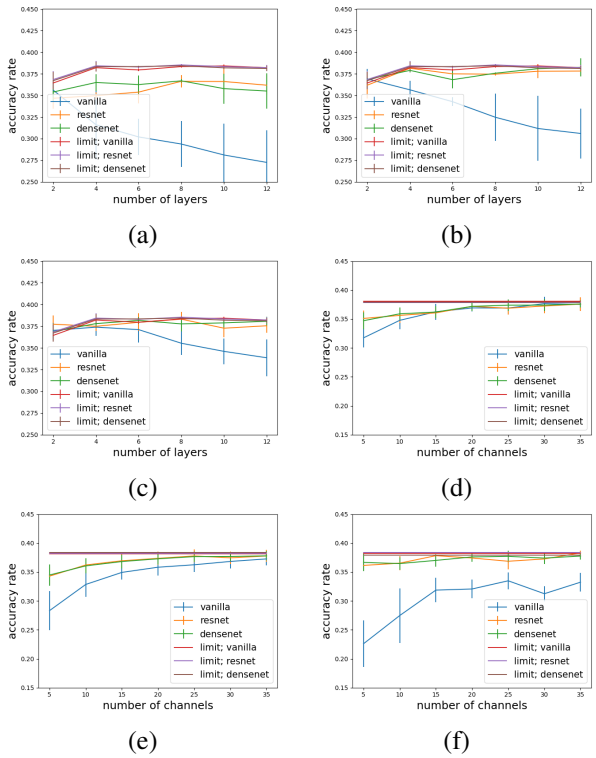



Figure 8: **Results on CIFAR10 for kernel regression over random gradient features.** We plot the averaged accuracy rates, when varying the width or the number of layers. In **(a-c)** we fix the width (number of channels) of $f(x; w)$ to be either **(a)** 8 **(b)** 16 or **(c)** 32, when varying the depth between 2-12. In **(d-f)** we fix the number of layers to be either **(d)** 3 **(e)** 6 or **(f)** 12, when varying the number of channels between 5-35. 'limit; vanilla', 'limit; resnet' and 'limit; densenet' stand for the results of the infinite-width limit kernels of vanilla networks, ResNets and DenseNets (resp.).


A. Golubeva, Behnam Neyshabur, and Guy Gur-Ari. Are wider nets better given the same number of parameters? *ArXiv*, abs/2010.14495, 2020.

B. Hanin and D. Rolnick. How to start training: The effect of initialization and architecture. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NeurIPS. Curran Associates, Inc., 2018.

Boris Hanin and Mihai Nica. Finite depth and width corrections to the neural tangent kernel. In *International Conference on Learning Representations*, ICLR, 2020. URL `https://openreview.net/forum?id=SJgndT4KwB`.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification, 2015.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

Gao Huang, Zhuang Liu, and Kilian Q. Weinberger. Densely connected convolutional networks. *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2261–2269, 2017.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning*, volume 37 of *ICML*, page 448–456. JMLR, 2015.

Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS, page 8580–8589, Red Hook, NY, USA, 2018. Curran Associates Inc.

Jaehoon Lee, Yasaman Bahri, Roman Novak, Samuel S. Schoenholz, Jeffrey Pennington, and Jascha Sohl-Dickstein. Deep neural networks as gaussian processes. In *International Conference on Learning Representations*, ICLR, 2018.

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. In *Advances in Neural Information Processing Systems 32*, pages 8572–8583. Curran Associates, Inc., 2019.

Etai Littwin, Tomer Galanti, Lior Wolf, and Greg Yang. On infinite-width hypernetworks. In *Advances in Neural Information Processing Systems 33*. Curran Associates, Inc., 2020a.

Etai Littwin, Ben Myara, Sima Sabah, J. Susskind, Shuangfei Zhai, and Oren Golan. Collegial ensembles. *ArXiv*, abs/2006.07678, 2020b.

Radford M. Neal. Priors for infinite networks. In *Bayesian Learning for Neural Networks*, volume 118 of *Lecture Notes in Statistics*. Springer, New York, NY, 1996.

Samuel Schoenholz, Justin Gilmer, Surya Ganguli, and Jascha Sohl-Dickstein. Deep information propagation. In *International Conference on Learning Representations*, ICLR, 11 2017.

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

Greg Yang. 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.

Greg Yang and Samuel S. Schoenholz. Mean field residual networks: On the edge of chaos. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS, page 2865–2873, Red Hook, NY, USA, 2017. Curran Associates Inc. ISBN 9781510860964.

Dingli Yu, Ruosong Wang, Zhiyuan Li, Wei Hu, Ruslan Salakhutdinov, Sanjeev Arora, and Simon S. Du. Enhanced convolutional neural tangent kernels, 2020. URL https://openreview.net/forum?id=BkgNqkHFPr.

Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In Edwin R. Hancock Richard C. Wilson and William A. P. Smith, editors, *Proceedings of the British Machine Vision Conference (BMVC)*, pages 87.1–87.12. BMVA Press, September 2016.

Hongyi Zhang, Yann N. Dauphin, and Tengyu Ma. Residual learning without normalization via better initialization. In *International Conference on Learning Representations*, ICLR, 2019. URL https://openreview.net/forum?id=H1gsz30cKX.