On the Random Conjugate Kernel and Neural Tangent Kernel

Zhengmian Hu [1] **Heng Huang** [1]

Abstract

We investigate the distributions of Conjugate Kernel (CK) and Neural Tangent Kernel (NTK) for ReLU networks with random initialization. We derive the precise distributions and moments of the diagonal elements of these kernels. For a feedforward network, these values converge in law to a log-normal distribution when the network depth d and width n simultaneously tend to infinity and the variance of log diagonal elements is proportional to d/n. For the residual network, in the limit that number of branches m increases to infinity and the width n remains fixed, the diagonal elements of Conjugate Kernel converge in law to a log-normal distribution where the variance of log value is proportional to $1/n$, and the diagonal elements of NTK converge in law to a log-normal distributed variable times the conjugate kernel of one feedforward network. Our new theoretical analysis results suggest that residual network remains trainable in the limit of infinite branches and fixed network width. The numerical experiments are conducted and all results validate the soundness of our theoretical analysis.

1. Introduction

Deep Neural Networks (DNNs) have been successfully applied to numerous applications such as computer vision (Krizhevsky et al., 2017; He et al., 2016), natural language processing (Bahdanau et al., 2014) and speech recognition (Graves et al., 2013), where DNNs often achieved superior performance compared to conventional machine learning algorithms and can also be used as a feature extractor (representation learning) for other methods.

[1]Department of Electrical and Computer Engineering, University of Pittsburgh, Pittsburgh, PA 15213, USA. This work was partially supported by NSF IIS 1845666, 1852606, 1838627, 1837956, 1956002, IIA 2040588.. Correspondence to: Zhengmian Hu <huzhengmian@gmail.com>, Heng Huang <heng.huang@pitt.edu>.

Proceedings of the 38^{th} International Conference on Machine Learning, PMLR 139, 2021. Copyright 2021 by the author(s).

Despite the extensive success, the DNN training process and mechanism of generalization are not fully understood due to the difficulties raised by the non-convexity of the loss function and the complication of optimization methods.

One way to simplify the DNNs analysis is to only train parameter in one layer while keep the parameters of other layers as fixed. Jarrett et al. (2009) demonstrated that only training last layer of a neural network can achieve competitive performance. Since all parameters of previous layers are fixed, the outputs at the last hidden layer $F(x, \theta)$ of non-linear DNNs are considered as a random feature vector whose inner product is known as Conjugate Kernel (CK) (Cho & Saul, 2009; Daniely et al., 2016; Poole et al., 2016; Schoenholz et al., 2016): $\Sigma(x, y) = F(x, \theta)^\top F(y, \theta)$.

The other way to simplify the DNNs analysis is using infinitesimal step size to conduct continuous time analysis (Arora et al., 2018; Du et al., 2019b; Su et al., 2014). The training process of parameters θ and neural network output $F(x, \theta)$ turns into two first order ordinary differential equations (ODE):

$$\frac{d\theta_i}{dt} = -\frac{\partial L}{\partial F(x,\theta)}\frac{\partial F(x,\theta)}{\partial \theta_i} \tag{1}$$

$$\frac{dF(y,\theta)}{dt} = \sum_i \frac{\partial F(y,\theta)}{\partial \theta_i}\frac{\partial F(x,\theta)}{\partial \theta_i}^\top \frac{\partial L}{\partial F(x,\theta)}. \tag{2}$$

The first ODE shows that the dynamics of parameters θ follows the gradient descent of a highly non-convex loss function $L(F(x, \theta))$. However, the dynamics of the output $F(y, \theta)$ enjoys the form of kernel gradient descent where the loss L is usually convex. The neural tangent kernel (NTK) is therefore defined as follows (Jacot et al., 2018):

$$K(x,y) = \sum_i \frac{\partial F(y,\theta)}{\partial \theta_i}\frac{\partial F(x,\theta)}{\partial \theta_i}^\top. \tag{3}$$

Both CK and NTK depend on the inputs and parameters of the neural network. They are potentially complicated due to the randomness of parameters at initialization and the changing of parameters during the training process.

One line of research bypass this complexity by conducting the analysis in the infinite-width limit with fixed depth. Neal

(2012); Williams (1997); Lee et al. (2018); de G. Matthews et al. (2018) showed that with proper scaling, a neural network with random initialization converges to a Gaussian process at infinite-width limit. Therefore all elements of conjugate kernel converge to a fixed value and CK is usually called as Neural Network Gaussian Process (NNGP) kernel.

The training dynamics under this regime also simplifies dramatically. Jacot et al. (2018) showed that as network width increases, the variance of NTK for a randomly initialized DNN reduces to zero and the NTK remains fixed during the training phase.

Another similar regime is to increase the depth to infinity at a slower rate than the network width. Under this regime, DNNs largely preserve the property of fixed CK and NTK. Huang et al. (2020) studied the limiting NTK of feedforward network and residual network by increasing widths to infinity first, and then increasing depths. Their result shows that the limiting NTK of feedforward network degenerates into delta function as the depth increases to infinity.

The fact that CK and NTK converge to fixed values at the infinite width limit reveals important properties about DNNs. First, it shows that the gradient descent optimizes wide DNNs to a global minimum. During the training of an infinite width neural network, the change of parameters reduces to zero and the network behaves like a linear model. The training dynamics of the output is precisely a kernel regression. With proper initialization and well-behaved loss function, the training loss will converge to zero. Second, it explains the generalization of an overparameterized network. NTK governs the generalization property of training an infinitely wide neural network, while CK is related to the generalization when we only train the last layer.

The limiting CK and NTK only depend on the network depth, the scaling of variance at each layer, and the choice of non-linearity function. The closed-form expression of CK and NTK allows directly evaluating an infinite width network and can be used for Kernel Regression or Kernel Support Vector Machines (Arora et al., 2019; 2020).

However, the success of DNNs has not been fully explained by the fixed CK and NTK. The NTK of an infinite width network generally has no feature learning capability, because it is data-independent. The training of DNNs, on the other hand, is mostly considered as a feature selection process. Arora et al. (2019); Chizat et al. (2019) demonstrate that the linearized DNNs with a fixed NTK could have a substantial gap of performance from finite width DNNs trained with gradient descent.

The above limitations of existing DNN analysis methods motivate us to study the prior distribution of CK and NTK for finite width DNNs. However, most existing works focus on controlling the deviation of finite width DNNs from the infinite width regime, rather than particular properties associated with finite width. This paper aims to study the CK and NTK for finite width network and the limiting behaviour when they converge to a random variable. The main contributions of this paper are summarized as follows:

- This paper provides new analysis for CK and NTK at finite width and depth condition. The precise distributions for diagonal elements of CK and NTK for every parameter are derived. Our results are applicable to both feedforward network and residual network with ReLU activation function.
- This paper derives upper bounds for every order moments of random CK and NTK, instead of only variance of them as in previous works. Our upper bounds on r-th order moment is proportional to $\exp(\binom{r}{2}\beta)$ where randomness coefficient β depends on network settings. The fast increasing high order moments demonstrate the long tail nature of CK and NTK for finite width network.
- Our new results can be generalized into infinite width and depth case, where we show that the limiting distribution of CK and NTK with random initialization are closely related to log-normal distribution. Our results endorse existing practice of modeling activation and gradient norm in neural network as log-normal distribution.
- We show that randomness coefficient β for residual network could remains bounded even in the limit of infinite depth and finite width. Therefore, an infinitely deep residual network with proper scaling is still trainable.

A comparison of properties of CK and NTK under different regimes of interest are shown in Figure 1.

2. Related Works

Convergence of Wide Network For infinite width neural networks, the NTK is a fixed value during training and the training trajectory is the same as kernel regression (Jacot et al., 2018; Lee et al., 2019). This is not true for any finite width network. However, neural networks with large but finite width still enjoy similar properties. More specifically, The NTK is almost fixed during training, and the training loss decreases to almost zero (Arora et al., 2019; Du et al., 2019a; Allen-Zhu et al., 2019; Buchanan et al., 2021). These works characterize the size of the deviations around the limiting behaviour in certain overparameterized regimes.

Finite Width Correction The research works along this direction focus on the deviation of NTK from its expectation without overparameterized assumption. Hanin & Nica (2020) provided an upper bound on the variance of diagonal element of NTK based on path counting. Littwin et al.

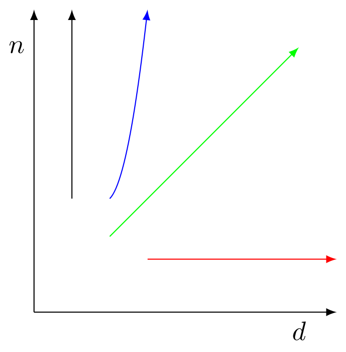


Figure 1: **Different regimes in the study of CK and NTK.** Network width and depth are denoted as d and n. Black line and Blue line: CK and NTK converge to fixed values. Green line: For feedforward network, diagonal elements of CK and NTK of each parameter converge to log-normal distribution. The variance of the log value is proportional to $\frac{d}{n}$. Diagonal elements of residual network NTK converges to the product of CK of a single feedforward network and a log normal distribution random variable. Red line: For feedforward network, the variance of CK and NTK is unbounded. For residual network, diagonal elements of CK converges to log-normal distribution, and diagonal elements of NTK converges in law to a log-normal distributed variable times CK of a feedforward network.

(2020) extended the result to networks with residual structure. These results do not reveal the heavy tail nature of prior distribution of CK and NTK, as only second order moments are discussed. Dyer & Gur-Ari (2020) presented bounds on general correlation functions in DNNs. However, most results are limited to linear network or remain to be conjectures.

Log-Normal Distribution in Neural Network Log-normal distribution has been used to approximate the output rate in neuroscience (Koulakov et al., 2009; Uzan et al., 2018). Hanin & Nica (2019); Gurbuzbalaban & Hu (2021) give theoretical guarantee that norm of feedforward network output converge to a log-normal distribution. In machine learning with artificial neural network, log-normal distribution is also observed to serve as a better approximation of gradient distribution than Gaussian distribution and inspires new technique for quantized training in (Chmiel et al., 2021).

3. Preliminary

Feedforward Networks We consider feedforward network with fully connected layers and ReLU activation. With assuming the input as x_0, the definition of each layer is:

$$\begin{aligned} y_i &= \sigma_i W_i x_{i-1} + b_i, i = 1, \dots, d \\ x_i &= ReLU(y_i), i = 1, \dots, d-1\,, \end{aligned} \tag{4}$$

where $f(x_0) = y_d$ is the output of the network and x_d is the output of last hidden layer. The length of vectors y_i and x_i is n_i. The parameters of the network are initialized to independent random variables $(W_i)_{j,k} \sim \mathcal{N}(0,1)$ and $b_i = 0$. Note that although b_i is set to 0 at initialization, this parameter is still trainable thus it contributes to the NTK.

We assume that output is a scalar and $n_d = 1$ while analyzing the NTK of feedforward network, and require output to be a vector with the same size as input x_0 when the feedforward network is used as one branch in residual network.

Residual Network We consider a residual network where every residual block is a feedforward network with ReLU activation. With input x_0, for $i = 0, \cdots, m-1$, we have:

$$\begin{aligned} x_{i,0} &= x_i \\ y_{i,j} &= \sigma_{i,j} W_{i,j} x_{i,j-1} + b_{i,j}\,, j = 1, \dots, d_i \\ x_{i,j} &= ReLU(y_{i,j}), j = 1, \dots, d_i - 1 \\ x_{i+1} &= x_i + y_{i,d_i} \\ y_{out} &= \sigma_{out} W_{out} x_m \end{aligned} \tag{5}$$

where m is the total number of residual branches, $f(x_0) = y_{out}$ is the output of the network. The length of vectors $y_{i,j}$ and $x_{i,j}$ is $n_{i,j}$. The length of vector x_i is $n_i = n_{i,0} = n_{i,d_i} = n$. The length of output y_{out} is $n_{out} = 1$. The parameters are initialized in a similar way as feedforward networks. The additional fully connected layer helps change the dimension of output and simplify the analysis of NTK.

Conjugate Kernel The conjugate kernel is defined as the inner product of last hidden layer. More specifically, CK is $\Sigma(x_0, \widetilde{x}_0) = x_{d-1}^T \widetilde{x}_{d-1}$ for feedforward networks and $\Sigma(x_0, \widetilde{x}_0) = x_m^T \widetilde{x}_m$ for residual network.

We also define the conjugate kernel of output as the inner of the output layer of the network. For feedforward networks $\Sigma'(x_0, \widetilde{x}_0) = y_d^T \widetilde{y}_d$ and for residual networks $\Sigma'(x_0, \widetilde{x}_0) = y_{out}^T \widetilde{y}_{out}$. These values arise in the analysis of residual network NTK.

Neural Tangent Kernel The NTK is defined as inner product of gradient with respect to network parameters. Therefore, NTK can be decomposed into summation of smaller NTKs, each of which comes from training a unique parameter in the network. More specifically,

$$\begin{aligned} K(x_0, \widetilde{x}_0) &= \sum_\theta K_\theta(x_0, \widetilde{x}_0) \\ K_\theta(x_0, \widetilde{x}_0) &= \frac{\partial f(x_0)}{\partial \theta} \frac{\partial f(\widetilde{x}_0)}{\partial \theta}^\top \end{aligned} \tag{6}$$

where θ is the weight or bias in one layer.

In this paper, we consider the case where $f(x_0)$ is a scalar, thus NTK can be described as a scalar as well. But our results can be easily extended to the case where a network has a vector output, and the NTK $K(x_0, \widetilde{x}_0)$ is a matrix.

Norm of Gaussian Random Vector We introduce two functions that will be used in our analysis. For an n-dimensional standard Gaussian vector $w \sim \mathcal{N}(0, I_{n\times n})$, we define the following two functions regarding to even order moments of w and $ReLU(w)$:

$$G_1(n,r) = \mathbf{E}\left[\|w\|^{2r}\right] = \prod_{t=0}^{r-1}(n+2t) \tag{7}$$

$$G_2(n,r) = \mathbf{E}\left[\|ReLU(w)\|^{2r}\right] = \frac{1}{2^n}\sum_{k=0}^{n}\binom{n}{k}G_1(k,r)$$

For large n, we have:

$$G_1(n,r) = n^r\left(1 + 2\binom{r}{2}\frac{1}{n} + O(\frac{1}{n^2})\right) \tag{8}$$

$$G_2(n,r) = \left(\frac{n}{2}\right)^r\left(1 + 5\binom{r}{2}\frac{1}{n} + O(\frac{1}{n^2})\right)$$

Log-Normal Distribution e^Z, where Z follows Gaussian distribution, is said to follow log-normal distribution. When $Z \sim \mathcal{N}(-\frac{\beta}{2}, \beta)$, we have the r-th order moment as $\mathbf{E}[(e^Z)^r] = \exp(\binom{r}{2}\beta)$. We use parameter β to characterize the randomness of such random variable. The log-normal distribution is heavy-tail in the sense that moment generating function $\mathbf{E}[\exp(te^Z)]$ is infinite for all positive t.

4. Feedforward Networks

We first introduce the precise distribution of diagonal elements of conjugate kernel.

Theorem 1 (CK of feedforward network). *The diagonal elements of conjugate kernel $\Sigma(x_0, x_0)$ have the same distribution as $\|x_0\|^2 \prod_{k=1}^{d-1}\sigma_k^2 \|ReLU(v_k)\|^2$ at initialization where $v_k \sim \mathcal{N}(0, I_{n_k\times n_k})$ is the independent standard Gaussian random vector.*

Moreover, $\Sigma'(x_0, x_0)$ has the same distribution as $\|x_0\|^2 \sigma_d^2 \|v_d\|^2 \prod_{k=1}^{d-1}\sigma_k^2 \|ReLU(v_k)\|^2$ at initialization.

We can further calculate the moments of CK as follows.

Corollary 1 (Moments of feedforward network CK). *The r-th order moment of CK of d-layer feedforward network with ReLU activation is:*

$$\mathbf{E}[\Sigma(x_0,x_0)^r] = \|x_0\|^{2r}\left(\prod_{k=1}^{d-1}\sigma_k^{2r}G_2(n_k,r)\right)$$
$$= \|x_0\|^{2r} c^r\left(\exp\left(\binom{r}{2}\beta\right) + \mathcal{O}(\sum_{i=1}^{d}\frac{1}{n_i^2})\right)$$

where r is a non-negative integer, $c = \prod_{k=1}^{d-1}\sigma_k^2\frac{n_k}{2}$ and $\beta = \sum_{k=1}^{d-1}\frac{5}{n_k}$.

We can derive the convergence of distribution as follows.

Theorem 2 (Limiting distribution of CK). *With the same definition of c and β as Corollary 1, at the limit of $n_1, \cdots, n_{d-1} \to \infty$ and c and β converge to fixed values, $c \to \widehat{c}$ and $\beta \to \widehat{\beta}$, the distribution of $\mathbf{E}[\Sigma(x_0,x_0)^r]$ converges to a log-normal distribution in law. More specifically, the limit distribution is the same as $\|x_0\|^2\widehat{c}e^Z$ where $Z \sim \mathcal{N}(-\frac{\widehat{\beta}}{2}, \widehat{\beta})$ is a Gaussian random variable.*

Remark 1. *By comparing Corollary 1 and moments of log-normal distribution, we can easily see that all moments converge to moments of log-normal distribution in the same limit in Theorem 2. However, we point out that the convergence in law as stated in Theorem 2 is stronger than convergence of all moments. The reason is that the long tail of log-normal distribution makes itself not determined by its moments (Heyde, 1963).*

Remark 2. *The results in Corollary 1 and Theorem 2 can be easily generalized to $\Sigma'(x_0, x_0)$. When we require n_d also increases to infinity, values c and β in Corollary 1 and Theorem 2 should be changed to $c = 2\prod_{k=1}^{d}\sigma_k^2\frac{n_k}{2}$, $\beta = \frac{2}{n_d} + \sum_{k=1}^{d-1}\frac{5}{n_k}$ and the distribution of $\Sigma'(x_0,x_0)$ still converges to log-normal distribution.*

When we have $n_d = 1$, the limiting distribution of $\Sigma'(x_0, x_0)$ is the same as $\|x_0\|^2\sigma_d^2\|w\|^2\widehat{c}e^Z$ where w is a standard Gaussian variable and other values are the same. Therefore, the activation of one single neuron doesn't converge to log-normal distribution.

Next, we derive the distribution of diagonal NTK coming from trainable weights W_i:

$$K_{W_i}(x_0, x_0) = \sum_{j,k}\frac{\partial f(x_0)}{\partial(W_i)_{j,k}}\frac{\partial f(x_0)}{\partial(W_i)_{j,k}}^\top. \tag{9}$$

This distribution is fully characterized by the connections between CK and NTK.

Theorem 3 (NTK of weights). *$K_{W_i}(x_0, x_0)$ at initialization has the same distribution as conjugate kernel $\sigma_d^2\Sigma(x_0, x_0)$.*

Remark 3. *For weights W_d at the last layer of feedforward network, $K_{W_d}(x_0, \widetilde{x}_0) = \sigma_d^2\Sigma(x_0, \widetilde{x}_0)$ always holds true. Theorem 3 shows that a similar relationship between NTK and CK exists for all other weights in the network when we only consider the distribution of diagonal elements at initialization.*

Remark 4. *Since the distribution of $K_{W_i}(x_0, x_0)$ is the same as $\sigma_d^2\Sigma(x_0, x_0)$, they also enjoy the same limiting behaviour. More specifically, at the limit of depth and width increasing to infinity at same rate, $K_{W_i}(x_0, x_0)$ also converges to a log-normal distribution. This supports the observation and assumption in (Chmiel et al., 2021) that gradient magnitudes are log-normal distributed.*

We then derive the distribution of diagonal NTK coming from trainable biases b_i.

$$K_{b_i}(x_0,x_0)=\sum_j \frac{\partial f(x_0)}{\partial (b_i)_j}\frac{\partial f(x_0)}{\partial (b_i)_j}^\top. \tag{10}$$

Theorem 4 (NTK of each biases)**.** *$K_{b_i}(x_0,x_0)$ has the same distribution as $\prod_{k=i}^{d-1}\sigma_{k+1}^2\left\|ReLU(v_k)\right\|^2$ where $v_k\sim\mathcal{N}(0,I_{n_k\times n_k})$.*

The moments are given by

$$\begin{aligned}\mathbf{E}[K_{b_i}(x_0,x_0)^r]&=\left(\prod_{k=i+1}^{d}\sigma_k^{2r}\right)\prod_{k=i}^{d-1}G_2(n_k,r)\\&=c^r\left(\exp\left(\binom{r}{2}\beta\right)+\mathcal{O}(\sum_{i=1}^{d}\frac{1}{n_i^2})\right)\end{aligned}$$

where $c=\prod_{k=i}^{d-1}\sigma_{k+1}^2\frac{n_k}{2}$ and $\beta=\sum_{k=i}^{d-1}\frac{5}{n_k}$.

This random variable also converges to a log-normal distribution in the limit that $n_i,\cdots,n_{d-1}\to\infty$ and c,β converge to fixed values.

We have derived the distributions and moments of weights NTKs and biases NTKs separately. However, the correlations between them are too convoluted to have a simple closed form solution. Therefore, we further adopt Jensen's inequality to derive an upper bound of any moments of the whole NTK:

$$K(x_0,x_0)=\sum_{i=1}^{d}(K_{W_i}(x_0,x_0)+K_{b_i}(x_0,x_0)). \tag{11}$$

Corollary 2 (The upper bound of moments of NTK)**.**

$$\frac{\mathbf{E}[K(x_0,x_0)^r]}{(\mathbf{E}[K(x_0,x_0)])^r}\le\exp\left(\binom{r}{2}\beta\right)+\mathcal{O}(\sum_{i=1}^{d}\frac{1}{n_i^2})$$

where $\beta=\sum_{k=1}^{d-1}\frac{5}{n_k}$.

By comparing the upper bound in Corollary 2 and moments of a log-normal distribution, we can conclude that the effective randomness coefficient is $\beta=\sum_{k=1}^{d-1}\frac{5}{n_k}$.

5. Residual Networks

We first give the distribution of diagonal CK of the residual networks.

Theorem 5 (CK of residual network)**.** *The conjugate kernel of m-branches residual network $\Sigma(x_0,x_0)$ has the same distribution as $\|x\|^2\prod_{i=0}^{m-1}\|\widehat{e}+V_i\|^2$ where $V_i=\sigma_{i,d_j}v_{i,d_i}\prod_{j=1}^{d_i-1}\sigma_{i,j}\left\|ReLU(v_{i,j})\right\|$, $\widehat{e}$ is a unit vector along any direction and $v_{i,j}\sim\mathcal{N}(0,I_{n_{i,j},n_{i,j}})$.*

Remark 5. *The random variable $\|V_i\|^2$ has the same distribution as $\Sigma'(\widehat{e},\widehat{e})$ of a single branch in residual network when regarded as a separate feedforward network.*

The moments of CK can be derived as follows. The closed form expression can be found at section A.1.

Corollary 3 (Moments of CK)**.** *We have*

$$\mathbf{E}[\Sigma(x_0,x_0)^r]=\exp\left(\left(r+\frac{4}{n}\binom{r}{2}\right)\sum_{i=0}^{m-1}c_i\right)+\mathcal{O}(\sum_{i=0}^{m-1}c_i^2)$$

where r is a positive integer, $c_i=2\prod_{k=1}^{d_i}\sigma_{i,k}^2\frac{n_{i,k}}{2}$.

Next, we provide the limiting distribution of CK of residual networks.

Theorem 6 (Limiting distribution of CK)**.** *Let $c_i=2\prod_{k=1}^{d_i}\sigma_{i,k}^2\frac{n_{i,k}}{2}$, $\beta_i=\frac{2}{n_{i,d_i}}+\sum_{k=1}^{d_i-1}\frac{5}{n_{i,k}}$ and $\beta=\frac{4}{n}$. At the limit of $c_i\to 0$, $\sum_{i=0}^{m-1}c_i\to\widehat{c}$ while β_i remains bounded and $\beta\to\widehat{\beta}$, the distribution of $\Sigma(x_0,x_0)$ converges in law to $e^{\widehat{c}}e^{Z}$, where $Z\sim\mathcal{N}(-\frac{\widehat{\beta}\widehat{c}}{2},\widehat{\beta}\widehat{c})$.*

Remark 6. *Theorem 6 is irrelevant to the specific shape of any residual branch, but only requires the randomness coefficients β_i for every branch to remain bounded. Therefore, this theorem is applicable to the residual network with infinite branches but every branch is a finite width and depth feedforward network.*

Remark 7. *The randomness coefficient of CK for residual network is $\frac{4}{n}\sum_{i=0}^{m-1}c_i$ and doesn't directly depend on depth. If only the summation of c_i is bounded, the randomness coefficient remains finite when depth increases to infinity. Therefore, the random feature at the last hidden layer of deep residual network is much stabler than that of a deep feedforward network.*

Remark 8. *Theorem 6 implies that in order to keep an arbitrarily deep residual network trainable, it is necessary that $\frac{4}{n}\sum_{i=0}^{m-1}c_i$ has an upper bound that is independent of branch number m. Similar phenomenon has been studied in previous works. Zhang et al. (2019) observed that if c_i of each layer is kept as a constant, the variance of network output will increase as branches increase. Arpit et al. (2019) proposed an initialization method that ensure $\sum_{i=0}^{m-1}c_i$ doesn't increase with m.*

We then derive the distribution of diagonal NTK of weights and biases.

Theorem 7 (NTK of each weights and biases)**.** *The diagonal NTK coming from weights in one layer $K_{W_{i,j}}(x_0,x_0)$ has the same distribution as $\|x\|^2\sigma_{out}^2\|V_i\|^2\prod_{s=0, s\neq i}^{m-1}\|\widehat{e}+V_s\|^2$. . The definition of V_i, $v_{i,j}$ and $\widehat{e}$ is the same as Theorem 5.*

The distribution of NTK coming from biases in one layer $K_{b_{i,j}}(x_0,x_0)$ is the same as

$\sigma_{out}^2 \left\|V'_{i,j}\right\|^2 \prod_{s=i+1}^{m-1} \|\widehat{e}+V_s\|^2$, where $V'_{i,j} = v_{i,d_i} \prod_{k=j}^{d_i-1} \sigma_{i,k+1} \|ReLU(v_{i,k})\|$.

We defer the closed form expression of NTK moments into Section A.1 due to the limit of space. Next, we derive the limiting distribution of NTKs and connect them with conjugate kernel of a feedforward network.

Theorem 8 (Limiting distribution of weights NTK)**.** *With the same distribution of c_i, β_i and β as Theorem 6, at the limit of $c_s \to 0$ for all $s \neq i$, $\sum_{\substack{s=0\\s\neq i}}^{m-1} c_s \to \widehat{c}$ while β_i remains bounded for all $s \neq i$ and $\beta \to \widehat{\beta}$, the random variable $K_{W_{i,j}}(x_0, x_0)$ converges in law to $\sigma_{out}^2 e^{\widehat{c}} e^Z \Sigma'_i(\widehat{e}, \widehat{e})$, where $\Sigma'_i(\widehat{e}, \widehat{e})$ is the conjugate kernel at the output layer of a feedforward network with the same shape as $(i+1)$-th branch in the residual network and $Z \sim \mathcal{N}(-\frac{\widehat{\beta}\widehat{c}}{2}, \widehat{\beta}\widehat{c})$.*

Theorem 9 (Limiting distribution of biases NTK)**.** *With the same distribution of c_i, β_i and β as Theorem 6, at the limit of $c_s \to 0$ for all $s > i$, $\sum_{s>i}^{m-1} c_s \to \widehat{c}$ while β_i remains bounded for all $s > i$ and $\beta \to \widehat{\beta}$, the random variable $K_{b_i}(x_0, x_0)$ converges in law to $\sigma_{out}^2 e^{\widehat{c}} e^Z \Sigma'_{i,j}(0, 0)$, where $Z \sim \mathcal{N}(-\frac{\widehat{\beta}\widehat{c}}{2}, \widehat{\beta}\widehat{c})$ and $\Sigma'_{i,j}(0,0)$ is the conjugate kernel at the output layer of a feedforward network with the same shape as $(i+1)$-th branch in the residual network but the bias b_j is initialized as a standard Gaussian vector.*

The above results on the weights NTK and biases NTK can also be combined to derive an upper bound on moments of the whole NTK.

Corollary 4 (The upper bound of the total NTK)**.**

$$\frac{\mathbf{E}[K(x_0,x_0)^r]}{(\mathbf{E}[K(x_0,x_0)])^r} \le \exp\left(\binom{r}{2}\left(\max_i \beta_i + \frac{4}{n}\sum_i c_i\right)\right) + \mathcal{O}\left(\sum_{i=0}^{m-1}\sum_{j=1}^{d_i}\frac{1}{n_{i,j}^2} + \sum_{i=0}^{m-1} c_i^2\right)$$

where $\beta_i = \frac{2}{n_{i,d_i}} + \sum_{k=1}^{d_i-1} \frac{5}{n_{i,k}}$ and $c_i = 2\prod_{k=1}^{d_i} \sigma_{i,k}^2 \frac{n_{i,k}}{2}$.

Remark 9. *By comparing the upper bound in Corollary 4 and moments of a log-normal distribution, we conclude that the effective randomness coefficient is $\max_i \beta_i + \frac{4}{n}\sum_i c_i$. Note that β_i only depends on the shape of a single branch, and effective randomness coefficient only depends on maximum β_i. This clearly contrasts with the feedforward network where, β increases linearly with depth. The extra term $\frac{4}{n}\sum_i c_i$ is the same as the randomness coefficient of CK. Therefore, with proper scaling, any order moments of NTK of residual network could remain bounded at infinite branches and finite width limit.*

6. Overview of Proof

Our analysis is based on the equivalent transform on random vectors and tensor networks. We note that many previous works adopt *path-based approach* to study neural network with ReLU activation (Hanin & Rolnick, 2018; Hanin & Nica, 2020; Littwin et al., 2020). This method decomposes the network output into a summation of all possible path connection input neuron to output one. Each path carries the weights and activation from all layers. Compared with this sum-over-path method, our method is more intuitive and reveals more properties of CK and NTK.

We first focus on the method for calculating CK and NTK of the feedforward network. After that, we generalize the method to residual network.

6.1. Conjugate Kernel

The last hidden layer of a feedforward ReLU network with biases initialized as 0 can be expressed as:

$$x_{d-1} = \prod_{i=1}^{d-1} \sigma_i ReLU(W_d(\dots(ReLU(W_1 x_0))\dots)).$$

We define an auxiliary random variable at each layer as $v_i = \frac{1}{\|x_{i-1}\|} W_i x_{i-1}$. If $\|x_{i-1}\| = 0$, we take v_i as a uniform random vector on the unit sphere. Then we have $v_i \sim \mathcal{N}(0, I_{n_i,n_i})$. Each layer of the feedforward network can be expressed as:

$$y_i = \sigma_i v_i \|x_{i-1}\|, x_i = \sigma_i ReLU(v_i)\|x_{i-1}\|. \quad (12)$$

Therefore, the norm of last hidden layer is $\|x_{d-1}\| = \|x_0\| \prod_{i=1}^{d-1} \sigma_i \|ReLU(v_i)\|$, and the norm of network output is $\|y_d\| = \|v_d\| \|x_0\| \prod_{i=1}^{d} \sigma_i \prod_{i=1}^{d-1} \|ReLU(v_i)\|$.

The output at each layer of the network y_i is dependent on the previous layer. However, we can show that the normalized output v_i is independent with each other.

Lemma 1. *All random vectors v_i for $i = 1, \dots, d$ are independent with each other.*

Based on above lemma, it is easy to prove the distribution of $\Sigma(x_0, x_0)$ and $\Sigma'(x_0, x_0)$ in Theorem 1.

6.2. Neural Tangent Kernel

In this section, we demonstrate our methods via proving the moments of NTK coming from weights. The proof is best shown in the language of tensor network diagram. Here we give a brief introduction of this notation, and we will give more examples to illustrate how the tensor graph notation corresponds to the standard vector and matrix notation in Section B.

We denote a vector v in diagram as v —. The node v in the diagram represents a tensor, and only one edge coming out of this node indicates that it is a rank-1 tensor or a vector. Similarly, a matrix V is represented as $-V-$. The contraction of tensors or matrix multiplication is represented as an edge connecting two tensors. For example for any two matrix A and B, $-A-B-$ represents the matrix multiplication AB. The identity matrix can be represented as an unbounded edge $-$ in the diagram without a node attached to it for simplicity. (Bridgeman & Chubb, 2017) provided a more comprehensive introduction on tensor network diagram. The main benefit of tensor network diagram is that trace, tensor product and tensor contraction can be expressed in a simple and unified way without extra notation.

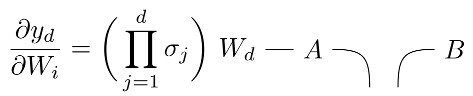
In the feedforward network with ReLU activation, $x_i = ReLU(y_i)$ can be written in a matrix multiplication form $x_i = D_i y_i$. The matrix D_i is a diagonal matrix where $[D_i]_{j,j}$ is 1 if $[y_i]_j > 0$ and is 0 if $[y_i]_j < 0$. When $[y_i]_j = 0$ we take $[D_i]_{j,j}$ as a Bernoulli random variable with $p = \frac{1}{2}$. Therefore the output of feedforward network can be written as $y_d = W_d D_{d-1} W_{d-1} \ldots D_1 W_1 x_0$. For the compactness of the diagram, we define $A = D_{d-1} W_{d-1} \ldots D_i$ and $B = D_{i-1} W_{i-1} \ldots D_1 W_1 x_0$ when we consider NTK coming from weights W_i in i-th layer.

With above definitions, the network output can be written as follows:

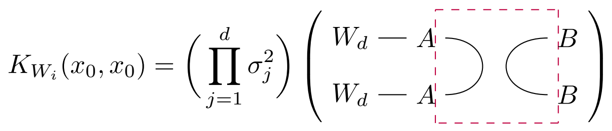
$$y_d = \left(\prod_{i=j}^{d} \sigma_j\right) W_d - A - W_i - B$$

The weights W_d at the last layer is a vector rather than matrix since we require scalar output y_d.

The partial gradient with respect to W_i can be represented as follows:



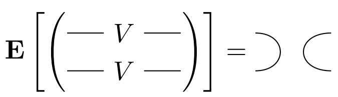
The NTK is just the inner product of gradients, therefore has the following form:



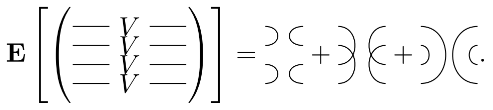
In the above diagram, we highlight the contraction in the red box. The key idea in our method is substituting this part of the tensor network with the expectation of another random tensor network.

Given a random matrix V where all elements are *i.i.d.* standard Gaussian variables. We have the expectation of $V \otimes V$, which is V tensor product with itself as $\mathbf{E} V_{i,j} V_{k,l} = I_{i,k} I_{j,l}$.

This can be represented in the following diagram:



This relationship can be generalized to higher order with Isserlis' theorem. For example:

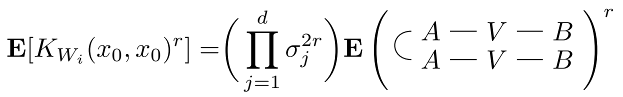


Generally, the expectation of $\otimes_{i=1}^{2r} V$, which is $2r$-th order tensor product of V to itself, can be represented as summation of $(2r-1)!!$ terms of tensor product of identity matrices, each of which corresponds to a pairing in a set of $2r$ elements.

Based on Isserlis' theorem, we can substitute the contraction part in the the diagram of $K_{W_i}(x_0, x_0)^r$ as follows:

$$K_{W_i}(x_0, x_0)^r = \frac{\prod_{j=1}^{d} \sigma_j^{2r}}{(2r-1)!!} \mathbf{E}_V \left(W_d - A - V - B \right)^{2r}$$

An extra term $(2r-1)!!$ shows up in the above result. We next take the expectation of W_d. Since elements in W_d is also *i.i.d.* standard Gaussian variables, the expectation of $\otimes_{i=1}^{2r} W_d$ also contains $(2r-1)!!$ terms of tensor product of identity matrices. Therefore, we have following result:



Recall that $\Sigma(x_0, x_0)^r = \left(\prod_{j=1}^{d-1} \sigma_j^{2r}\right) \|AW_iB\|^{2r}$. By using similar auxiliary random variable in section 6.1, we have $\mathbf{E}[K_{W_i}(x_0, x_0)^r] = \sigma_d^{2r} \mathbf{E}[\Sigma(x_0, x_0)^r]$.

For a network with only one hidden layer, the distribution of $\Sigma(x_0, x_0)$ is determined by its moments according to Carleman's condition. Therefore, the distributions of diagonal weights NTK and diagonal CK are same. For feedforward network with more than one hidden layer, we need to set up auxiliary variable in backward manner, the specific proof is shown in Section D. The biases NTK can be controlled in similar way.

6.3. Residual Network

For conjugate kernel, we can adopt auxiliary random variables $v_{i,j}$ similar to Section 6.1. We only need to show $v_{i,j}$ are independent with each other to prove Theorem 5.

The NTK can be controlled with same methods in Section 6.2. The only difference is that when we calculate the NTK of parameter in one branch, we need to remove the skip connection bypassing that branch in the tensor network.

Moment	Predicted	Estimated
$\mathbf{E}\Sigma(x_0,x_0)^1$	2.50E-1	2.52E-01 $\pm$ 2.55E-03
$\mathbf{E}\Sigma(x_0,x_0)^2$	6.89E-2	6.98E-02 $\pm$ 1.49E-03
$\mathbf{E}\Sigma(x_0,x_0)^3$	2.08E-2	2.13E-02 $\pm$ 7.43E-04
$\mathbf{E}\Sigma(x_0,x_0)^4$	6.86E-3	7.08E-03 $\pm$ 3.73E-04

Table 1: Verification of CK and NTK of a three layers feedforward network.

Moment	Predicted	Estimated
$\mathbf{E}\Sigma(x_0,x_0)^1$	2.25E+0	2.25E+00 $\pm$ 4.09E-03
$\mathbf{E}\Sigma(x_0,x_0)^2$	5.23E+0	5.23E+00 $\pm$ 1.95E-02
$\mathbf{E}\Sigma(x_0,x_0)^3$	1.26E+1	1.26E+01 $\pm$ 7.33E-02
$\mathbf{E}\Sigma(x_0,x_0)^4$	3.13E+1	3.12E+01 $\pm$ 2.58E-01

Table 2: Verification of CK and NTK of a two branches residual network. Every branch is a feedforward network with one hidden layer.

7. Experimental Results

In this section, we will validate our theoretical results with numerical experiments.

7.1. Verifying the Moments

We sample 1000 three layers feedforward network and calculate the NTK and CK at initialization. All parameters are initialized in the distribution specified in Section 3. The hyper-parameters are set as $n_0 = n_1 = n_2 = 100$, $n_2 = 1$ and $\sigma_i = \frac{1}{10}$. The input x_0 is sampled from a unit sphere.

The theoretical prediction and estimation obtained in numerical experiment is shown in Table 1. We only show the moments for $\Sigma(x_0, x_0)$ in this section. Moments of $\Sigma'(x_0, x_0)$ and NTKs are deferred to Section C.1 due to the limit of space. We don't include higher order moments because the long tail nature of CK distribution makes it harder to numerically estimate higher order moments. The theory and experiment results fit well in terms of first 4 order moments, which demonstrates the accuracy of our theoretical results.

We also compare the moments of residual network in Table 2 by sampling 10000 residual network with $n_{i,j} = 100$ and $\sigma_{i,j} = \frac{1}{10}$. Only CK moments are shown below and the full result can be found in Section C.1.

7.2. Verifying the Log-Normal Distribution

In the following experiment, we will verify our theoretical results on limiting distribution of diagonal CK and NTK.

According to Theorem 2, the diagonal CK of feedforward network converges to the log-normal distribution in the infinite width and infinite depth limit. We demonstrate the convergence in Figure 2, in which we take three feedforward networks with different depth. The hyper-parameters including σ_i and n_i are tuned to keep $c = 1$ and $\beta \approx 1$. We reinitialize these networks 10^6 times and plot the empirical distribution of CK. Figures of empirical distribution for other settings can be found in section C.2.

We use Kolmogorov–Smirnov test to calculate the difference between the distributions of CK and NTK and their limiting distributions to validate Theorems 2 to 4. Table 3 shows

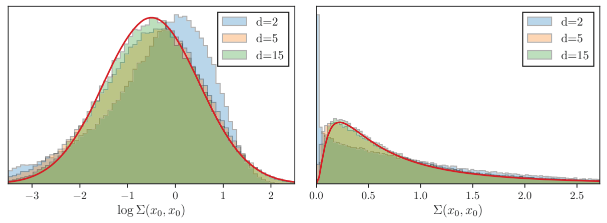


Figure 2: **Distribution of CK** The red line is the theoretical limiting distribution given c and β.

that the p-value for CK and NTK coming from weights and biases at first layer. The KS test takes 1000 samples of CK and NTK can calculate the maximum difference between the empirical distribution and the theoretical limiting distribution. The hyper-parameters are the same as neural networks used in Figure 2. Table 3 illustrates that when $d = 2$, we have strong evidence to support that the distributions of CK and NTK are not log-normal. However, when the depth increases to 5 and 15, it is hard to distinguish them from the log-normal distribution.

We further verify the limiting behaviour in Theorems 8 and 9. For CK of residual network, we test the empirical distribution of $\Sigma(x_0, x_0)$ against the limiting the log-normal distribution. The NTK of residual network is tested against another sample of $e^Z\Sigma(x_0, x_0)$ where $\Sigma(x_0, x_0)$ is CK of a finite sized feedforward network and e^Z is independently sampled from the log normal distribution. The sample size in Table 4 is also 1000. We set the branch of residual network as the feedforward networks with one hidden layer and hyper-parameters are tuned so that $\sum_i c_i = 1$ and $\beta = \frac{4}{100}$.

We note that the p-value for conjugate kernel in Table 4 gradually increases, which indicates the distribution of CK gradually converges to the log-normal distribution. The p-value for NTK in a residual network with only one branch is very high. This is expected because the variance of log-normal distribution coming from other branches is 0.

We also conduct empirical study on the correlation between NTK from different parameters in Section C.3. Although the joint probability is complicated, we can provide approximation on the correlation coefficient of logarithm NTKs.

Depth	$\Sigma(x_0, x_0)$	$K_{W_1}(x_0, x_0)$	$K_{b_1}(x_0, x_0)$
2	1.60E-10	3.48E-09	3.48E-09
5	1.30E-01	1.44E-02	1.44E-02
15	7.13E-01	1.23E-01	1.23E-01

Table 3: The p-value of KS test on CK and NTK of the feedforward network.

Branches	$\Sigma(x_0, x_0)$	$K_{W_{0,1}}$	$K_{b_{0,1}}$
1	0.00E+00	8.28E-01	5.73E-01
5	1.65E-31	1.62E-04	3.96E-05
20	3.97E-05	7.76E-02	2.41E-01
100	1.36E-01	9.71E-02	1.34E-01

Table 4: The p-value of KS test on CK and NTK of the residual network.

The statistics property of non diagonal elements of random CK and NTK are discussed in Section C.4. In Section C.5, the random gradient are used as random feature for classification on UCI dataset.

8. Conclusion

We derive the precise distributions and moments of diagonal elements of Conjugate Kernel (CK) and Neural Tangent Kernel (NTK) for ReLU networks. These distributions have long tail similar to log-normal distribution and can be characterized by one parameter β. We show that randomness coefficient β has different dependency on the network depth and width in different architecture. For residual network, β could remain bounded even in the infinite depth and finite width limit.

This paper includes two regimes where CK and NTK converge to a distribution. In contrast to infinite width and fixed depth limit where NTK is fixed, our results show the possibility of a neural network to learn data-dependent feature. However, the dynamics of NTK during training still largely remains unknown, as the training process is complicated and data-dependent. We believe explaining how NTK achieves better alignment with data during training is a challenging but appealing direction.

References

Allen-Zhu, Z., Li, Y., and Song, Z. A convergence theory for deep learning via over-parameterization. In *International Conference on Machine Learning*, pp. 242–252. PMLR, 2019.

Arora, S., Cohen, N., and Hazan, E. On the optimization of deep networks: Implicit acceleration by overparameterization. In *International Conference on Machine Learning*, pp. 244–253. PMLR, 2018.

Arora, S., Du, S. S., Hu, W., Li, Z., Salakhutdinov, R., and Wang, R. On exact computation with an infinitely wide neural net. In *Thirty-third Conference on Neural Information Processing Systems*, 2019.

Arora, S., Du, S. S., Li, Z., Salakhutdinov, R., Wang, R., and Yu, D. Harnessing the power of infinitely wide deep nets on small-data tasks. In *International Conference on Learning Representations*, 2020.

Arpit, D., Campos, V., and Bengio, Y. How to initialize your network? robust initialization for weightnorm &amp; resnets. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

Bahdanau, D., Cho, K., and Bengio, Y. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

Bridgeman, J. C. and Chubb, C. T. Hand-waving and interpretive dance: an introductory course on tensor networks. *Journal of Physics A: Mathematical and Theoretical*, 50 (22):223001, 2017.

Buchanan, S., Gilboa, D., and Wright, J. Deep networks and the multiple manifold problem. In *International Conference on Learning Representations*, 2021.

Chizat, L., Oyallon, E., and Bach, F. On lazy training in differentiable programming. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 32, pp. 2937–2947, 2019.

Chmiel, B., Ben-Uri, L., Shkolnik, M., Hoffer, E., Banner, R., and Soudry, D. Neural gradients are near-lognormal: improved quantized and sparse training. In *International Conference on Learning Representations*, 2021.

Cho, Y. and Saul, L. Kernel methods for deep learning. In Bengio, Y., Schuurmans, D., Lafferty, J., Williams, C., and Culotta, A. (eds.), *Advances in Neural Information Processing Systems*, volume 22, pp. 342–350. Curran Associates, Inc., 2009.

Daniely, A., Frostig, R., and Singer, Y. Toward deeper understanding of neural networks: The power of initialization and a dual view on expressivity. In Lee, D., Sugiyama, M., Luxburg, U., Guyon, I., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 29, pp. 2253–2261. Curran Associates, Inc., 2016.

de G. Matthews, A. G., Hron, J., Rowland, M., Turner, R. E., and Ghahramani, Z. Gaussian process behaviour in wide deep neural networks. In *International Conference on Learning Representations*, 2018.

Du, S., Lee, J., Li, H., Wang, L., and Zhai, X. Gradient descent finds global minima of deep neural networks. In *International Conference on Machine Learning*, pp. 1675–1685. PMLR, 2019a.

Du, S. S., Zhai, X., Poczos, B., and Singh, A. Gradient descent provably optimizes over-parameterized neural networks. In *International Conference on Learning Representations*, 2019b.

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

Graves, A., Mohamed, A.-r., and Hinton, G. Speech recognition with deep recurrent neural networks. In *2013 IEEE international conference on acoustics, speech and signal processing*, pp. 6645–6649. IEEE, 2013.

Gurbuzbalaban, M. and Hu, Y. Fractional moment-preserving initialization schemes for training deep neural networks. In *International Conference on Artificial Intelligence and Statistics*, pp. 2233–2241. PMLR, 2021.

Hanin, B. and Nica, M. Products of many large random matrices and gradients in deep neural networks. *Communications in Mathematical Physics*, pp. 1–36, 2019.

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

Hanin, B. and Rolnick, D. How to start training: The effect of initialization and architecture. *arXiv preprint arXiv:1803.01719*, 2018.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Heyde, C. C. On a property of the lognormal distribution. *Journal of the Royal Statistical Society: Series B (Methodological)*, 25(2):392–393, 1963.

Huang, K., Wang, Y., Tao, M., and Zhao, T. Why do deep residual networks generalize better than deep feedforward networks?–a neural tangent kernel perspective. *arXiv preprint arXiv:2002.06262*, 2020.

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.

Jarrett, K., Kavukcuoglu, K., Ranzato, M., and LeCun, Y. What is the best multi-stage architecture for object recognition? In *2009 IEEE 12th international conference on computer vision*, pp. 2146–2153. IEEE, 2009.

Koulakov, A. A., Hromádka, T., and Zador, A. M. Correlated connectivity and the distribution of firing rates in the neocortex. *Journal of Neuroscience*, 29(12):3685–3694, 2009.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017.

Lee, J., Sohl-dickstein, J., Pennington, J., Novak, R., Schoenholz, S., and Bahri, Y. Deep neural networks as gaussian processes. In *International Conference on Learning Representations*, 2018.

Lee, J., Xiao, L., Schoenholz, S., Bahri, Y., Novak, R., Sohl-Dickstein, J., and Pennington, J. Wide neural networks of any depth evolve as linear models under gradient descent. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 32, pp. 8572–8583, 2019.

Littwin, E., Galanti, T., and Wolf, L. On random kernels of residual architectures. *arXiv e-prints*, pp. arXiv–2001, 2020.

Neal, R. M. *Bayesian learning for neural networks*, volume 118. Springer Science & Business Media, 2012.

Poole, B., Lahiri, S., Raghu, M., Sohl-Dickstein, J., and Ganguli, S. Exponential expressivity in deep neural networks through transient chaos. In Lee, D., Sugiyama, M., Luxburg, U., Guyon, I., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 29, pp. 3360–3368. Curran Associates, Inc., 2016.

Schoenholz, S. S., Gilmer, J., Ganguli, S., and Sohl-Dickstein, J. Deep information propagation. In *International Conference on Learning Representations*, 2016.

Su, W., Boyd, S. P., and Candes, E. J. A differential equation for modeling nesterov's accelerated gradient method: Theory and insights. In *NIPS*, volume 14, pp. 2510–2518, 2014.

Uzan, H., Sardi, S., Goldental, A., Vardi, R., and Kanter, I. Stationary log-normal distribution of weights stems from spontaneous ordering in adaptive node networks. *Scientific reports*, 8(1):1–9, 2018.

Williams, C. K. Computing with infinite networks. *Advances in neural information processing systems*, pp. 295–301, 1997.

Zhang, H., Dauphin, Y. N., and Ma, T. Fixup initialization: Residual learning without normalization. 2019.