Fractional moment-preserving initialization schemes for training deep neural networks

Mert Gürbüzbalaban
mg1366@rutgers.edu
Department of Management
Science and Information Systems
Rutgers Business School
Piscataway, NJ 08550

Yuanhan Hu
yuanhan.hu@rutgers.edu
Department of Management
Science and Information Systems
Rutgers Business School
Piscataway, NJ 08550

Abstract

A traditional approach to initialization in deep neural networks (DNNs) is to sample the network weights randomly for preserving the second moment of layer outputs. On the other hand, recent results show that training with SGD can result in heavy-tailedness in the distribution of the network weights with a potentially infinite variance. This suggests that the traditional approach to initialization may be restrictive as SGD updates do not necessarily preserve the finiteness of the variance of layer outputs. Motivated by this, we develop initialization schemes for fully connected feed-forward networks that can provably preserve any given moment of order $s \in (0, 2]$ over the layers for a class of activations including ReLU, Leaky ReLU, Randomized Leaky ReLU and linear activations. These generalized schemes recover traditional initialization schemes in the limit $s \to 2$ and serve as part of a principled theory for initialization. For all these schemes, we show that the network output admits a finite almost sure limit as the number of layers grows, and the limit is heavy-tailed in some settings. We also prove that the logarithm of the norm of the network outputs, if properly scaled, will converge to a Gaussian distribution with an explicit mean and variance we can compute depending on the activation used, the value of s chosen and the network width, where log-normality serves as a further justification of why the norm of the network output can be heavy-tailed in DNNs. We also prove that our initialization scheme avoids small network output values more frequently compared to traditional approaches. Our results extend if dropout is used and the proposed initialization strategy does not have an extra cost during the training procedure. Finally, we discuss extensions of our results to convolutional neural networks and show through numerical experiments that our initialization can improve the initial stages of training.

Proceedings of the 24th International Conference on Artificial Intelligence and Statistics (AISTATS) 2021, San Diego, California, USA. PMLR: Volume 130. Copyright 2021 by the author(s).

1 Introduction

Initialization of the weights of a deep neural network (DNN) plays a crucial role on the training and test performance (Daniely et al., 2016; Hanin and Rolnick, 2018; Sutskever et al., 2013) where random weight initialization often yields a favorable starting point for optimization (Daniely et al., 2016). A common traditional approach to initialization that goes back to 1990's is to initialize the weights randomly in a way to preserve the variance of the output of each network layer (LeCun et al., 1998b; Bottou, 1988) which avoids the network to reduce or magnify the norm of the input signal exponentially. For fully-connected networks with a fixed number of neurons d at each layer with linear activations, this can be achieved by setting the bias vectors to zero and sampling the weights in an independent and identically distributed (i.i.d.) fashion from a Gaussian or uniform distribution with mean zero and variance $\sigma^2 = 1/d$ (LeCun et al., 1998b), proposed originally for tanh activations (Kalman and Kwasny, 1992). This initialization is referred to as "Lecun initialization" in the literature. More recently, He et al. (2015) showed that the choice of $\sigma^2 = \frac{2}{d}$ keeps the variance constant if ReLU activation is used instead; where the extra factor of 2 is to account for the fact that ReLU output is zero with probability 1/2 when

input is a mean-zero symmetric distribution without an atom at zero. This initialization is sometimes referred to as "Kaiming initialization" in the literature (Luther and Seung, 2019). A similar initialization rule that can preserve the variance for parametric ReLU and Leaky ReLU activations are also developed in He et al. (2015), where parametric ReLU and Leaky ReLU are variants of ReLU, proposed to improve the performance of ReLU (Maas et al., 2013).

On the other hand, recent research shows that during the training process with stochastic gradient descent (SGD), the distribution of stochastic gradients can become heavy-tailed over time with a non-Gaussian behavior even though in the initial iterations, stochastic gradients may have a Gaussian-like behavior with a finite variance (Şimşekli et al., 2019b,a; Gürbüzbalaban et al., 2020), where heavy tailedness of a distribution refers to the fact that its tail is heavier than an exponential distribution (Foss et al., 2011). In this setting, based on the empirical distribution of stochastic gradients, modelling stochastic gradients with an α-stable distribution has been proposed (Şimşekli et al., 2019b,a), which is a distribution that does not have a finite variance but rather has a (fractional) moment of order s satisfying $s < \alpha < 2$. This heavy-tailed behavior in the stochastic gradients is also naturally inherited by the network weights due to SGD updates and the amount of heavy tail is also related to the batchsize (Panigrahi et al., 2019; Gürbüzbalaban et al., 2020). Heavy tails for SGD have also been observed in Zhang et al. (2019) and have been associated with better generalization (Martin and Mahoney, 2019; Şimşekli et al., 2020). In Martin and Mahoney (2019), modelling weights of a well-optimized neural network with a "Pareto distribution" with shape paremeter λ is proposed, which is another heavy-tailed distribution with a power law tail (Resnick, 2007) and an infinite variance when $\lambda < 2$ but with a finite s-th moment for any $s \in (0, \lambda)$. These results regarding the heavy-tailedness of the network weights with a potentially infinite variance suggest that the traditional approach of preserving the second moment and variance of layer outputs at the initialization level may be restrictive as SGD updates do not necessarily preserve the finiteness of the variance after all. This raises the question whether more general initialization schemes that can preserve the s-th moment can be developed for a given $s \in (0, 2]$ rather than the traditional case which covers only $s = 2$.

Contributions. In this paper, we develop a novel class of initialization schemes that can preserve a fractional moment of order $s \in (0, 2]$ over the layer outputs during the forward pass. The schemes are applicable to ReLU, parameteric ReLU, Leaky ReLU, Randomized Leaky ReLU and linear activations for fully-connected deep neural networks. We then provide experiments to show that our schemes acts as a warm start for SGD in the sense that it improves the training and test accuracy in the initial stages of training over the MNIST and CIFAR-10 datasets compared to traditional initializations. The main idea behind our initialization is to initialize the network weights as i.i.d. Gaussian variables $\sim \mathcal{N}(0, \sigma^2)$ but adjust the variance σ^2 in a special way as a function of s to keep the s-th moment invariant during the forward pass. To our knowledge, the choice of σ^2 that can preserve the s-th moment of layer output vectors has not been studied in the literature before our work. For this purpose, first we develop analytical formulas that express the s-th moment of the k-th layer output for any $s \in (0, 2)$ and in any dimension d for the ReLU, Leaky ReLU, Randomized Leaky ReLU and linear activations (Theorems 1, 7). Our proof relies on adapting the techniques of Cohen and Newman (1984) developed for the products of random matrices with i.i.d. Gaussian entries to nonlinear stochastic recursions arising in forward propagation with nonlinear activations and exploiting the piecewise linear structure of ReLU and parametric ReLU activations. This yields explicit formulas regarding how to choose the initialization weight variance σ^2 to preserve the s-th moment (Corollary 4, 10). Our initialization scheme allows to choose a larger σ^2 compared to Kaiming initialization, and is the main reason why with our initialization scheme, network outputs small values relatively less frequently so that small gradients occur less frequently at the initialization. In fact, we show that the logarithm of the norm of the network outputs, if properly scaled, will converge to a Gaussian distribution with an explicit mean and variance we can compute as the number of layers grows (Theorem 5, 11), where log-normality serves as a further theoretical justification of why the norm of the network output can be heavy-tailed in DNNs. Such a log-normality result was previously shown in Hanin and Nica (2019) (see also (Hanin, 2018)) for ReLU and linear activations in the regime where the width and depth of the network simultaneously tend to infinity, when the weights are initialized from an arbitrary symmetric distribution with fourth moments; however explicit formulas for the asymptotic mean and variance were not given for the finite width regime. Our results are explicit for finite width and are also applicable to parametric ReLU and Leaky ReLU activations, enabling us to show that if the number of layers is sufficiently large, our scheme will have a first-order stochastic dominance property over the traditional Kaiming initialization in the sense of Hadar and Russell (1969) (see Remarks 6 and 12). Intuitively speaking, the cumulative distribution function (cdf) of the norm of the network output with our initialization will be strictly shifted to the right compared to the cdf of Kaiming initialization

(see Figure 2) and therefore will avoid taking smaller values more often. With zero bias vectors and fixed width over layers, we show that L_p and almost sure limits of network outputs can be only zero or infinity depending on whether σ exceeds an explicit threshold we provide (Theorem 14). If additive noise is added to post-activations, we show that the almost sure limit of output layers is heavy-tailed for linear activations. The results show that forward pass can make the network output and (hence the gradient of the training cost) heavy-tailed if the variance of network weights exceed a certain threshold, even if the weights are i.i.d. Gaussian (Theorem 15), shedding further light into the origins of heavy tails during signal propagation in DNNs. Our results extend if dropout (Srivastava et al., 2014) is used (Remark 13). Also, our framework recovers a number of traditional initialization schemes such as Lecun initialization and Kaiming initialization in the limit as $s \to 2$, and therefore serves as a principled theory for initialization. Furthermore, our results extend naturally to convolutional neural networks, which we discuss in the appendix due to space considerations.

Related literature. There are alternative approaches to initialization based on taking an average of the width of input and output layers to balance off efficient forward propagation with backward propagation (Glorot and Bengio, 2010; Defazio and Bottou, 2019). In this paper, we consider forward propagation, but backward propagation analysis is almost the same for ReLU and Leaky ReLU activations by simply replacing the number of input layers with number of output layers in the analysis (see e.g. (He et al., 2015; Glorot and Bengio, 2010; Defazio and Bottou, 2019)) and our initialization schemes can in principle be combined with such averaging strategies. There are also many other strategies that enhance signal propagation in deep networks such as orthogonal matrix initialization (Saxe et al., 2013), random walk initialization (Sussillo and Abbott, 2014), edge of chaos initialization (Yang and Schoenholz, 2017; Hayou et al., 2018; Schoenholz et al., 2016) and mean field theory based approaches (Xiao et al., 2018; Blumenfeld et al., 2019), batch normalization (Ioffe and Szegedy, 2015), composition kernels (Daniely et al., 2016), approaches for residual networks (Yang and Schoenholz, 2017; Hanin and Rolnick, 2018; Ling and Qiu, 2019) as well as development of alternative activation functions (Klambauer et al., 2017; Clevert et al., 2015; Hayou et al., 2018) and automating the search for good initializations (Dauphin and Schoenholz, 2019).

Notation. We use standard notation, common in the machine learning literature; however we provide a detailed discussion of the notation used in our paper in the supplementary material (Appendix A).

2 Preliminaries and Setting

Fully connected feed-forward networks and activation functions. We consider a fully connected feed-forward deep neural network. Given input data $x^{(0)} \in \mathbb{R}^d$, these networks consist of multiple layers. The output of the k-th layer which we denote by $x^{(k)}$ follows the following recursion:

$$x^{(k+1)} = F^{(k+1)}(x^{(k)}),$$
$$F^{(k+1)}(x) := \phi_a(W^{(k+1)}x + b^{(k+1)}),$$

where $W^{(k+1)} \in \mathbb{R}^{d\times d}$ and $b^{(k+1)} \in \mathbb{R}$ are the weight matrix and the bias of the $(k+1)$-st layer respectively and the function ϕ_a denotes the parametric ReLU activation function (He et al., 2015) applied componentwise to a vector, defined for a scalar input $z \in \mathbb{R}$ as

$$\phi_a(z) = \begin{cases} z & \text{if } z > 0, \\ az & \text{if } z \leq 0, \end{cases} \tag{2.1}$$

where $a \in [0, 1]$ is a parameter. The parameter a can also be learned from data during training (He et al., 2015), but in this paper we are interested in the case where the choice of a will be given and fixed. Depending on the choice of a, this class recovers a number of activation functions of interest:

1. For $a = 0$, $\phi_0(x) = \max(0, x)$ is the rectified linear unit (ReLU) which is widely used in practice (Maas et al., 2013).
2. For $a = 0.01$, this corresponds to Leaky ReLU activation (Maas et al., 2013). More recently, some other choices of $a \in (0, 1)$ has also been considered (He et al., 2015). If a is chosen randomly, this is referred to as Randomized Leaky ReLU (Xu et al., 2015).
3. For $a = 1$, $\phi_1(x) = x$ is the linear activation function.

Gaussian initialization techniques. We consider Gaussian initialization where the network weights are independent and identically distributed (i.i.d) following a Gaussian distribution with constant variance σ^2 and mean zero and biases are set to zero, i.e. we assume:

(A1) All the weights are independent and identically distributed (i.i.d.) with a centered Gaussian distribution satisfying $W^{(k)} \in \mathbb{R}^{d\times d}$, $W^{(k)}_{ij} \sim \mathcal{N}(0, \sigma^2)$ for every i,j and k where $\sigma^2 > 0$ is the variance of the k-th layer with width d.

(A2) The biases are initialized to zero, i.e. $b^{(k)} = 0$ for every $k \geq 1$.

For simplicity of the presentation, above we assume that the width of the network is equal to d and is constant over different layers. However, our results naturally extends to the case if each layer k has a different width d_k (see Remark 3). The popular Kaiming initialization corresponds to the choice of $\sigma^2 = \frac{2}{d(1+a^2)}$ which preserves the second moment of the layer outputs, we will next show that there exists a *critical variance* level $\bar{\sigma}_a^2(s,d)$ that we can compute explicitly, so that the choice of $\sigma^2 = \bar{\sigma}_a^2(s,d)$ will preserve the s-th moment of the output over the layers in any dimension d for any $s \in (0,2]$ given. We start with the ReLU case which corresponds to $a = 0$.

3 ReLU Activation

In the next result, we characterize arbitrary moments of the output of the k-th layer, i.e. we provide an explicit formula for $\mathbb{E}(\|x^{(k)}\|^s)$ where $s > 0$ can be any real scalar where (throughout this paper) and $\|\cdot\|$ denotes the Euclidean (L_2) norm. Our result identifies three regimes: For given width d and moment $s > 0$, there exists a threshold $\bar{\sigma}_0(s,d)$ for choosing the standard deviation σ of the initialization: If we choose $\sigma = \bar{\sigma}_0(s,d)$, then the network with ReLU activation will preserve the s-th moment. The choice of σ below (resp. above) this threshold, will lead to s-th moment to decay (resp. grow) exponentially fast. The result relies on expressing the output of the layers as a mixture of chi-square distributions with binomial mixture weights based on adaptations of the techniques from Cohen and Newman (1984) from linear stochastic recursions to the nonlinear case. The proof of this result, and the proof of all the other results, can be found in the supplementary material.

Theorem 1. ***(Explicit characterization of the critical variance*** $\bar{\sigma}_0^2(s,d)$***)*** *Consider a fully connected network with an input $x^{(0)} \in \mathbb{R}^d$ and Gaussian initialization satisfying **(A1)-(A2)** with ReLU activation function $\phi_0(x) = \max(x,0)$. Let $s > 0$ be a given real scalar. The s-th moment of the output of the k-th layer is given by*

$$\mathbb{E}\left[\|x^{(k)}\|^s\right] = \|x^{(0)}\|^s(\sigma^s I_0(s,d))^k, \tag{3.1}$$

$$I_0(s,d) = 2^{s/2}\sum_{n=0}^{d}\binom{d}{n}\frac{1}{2^d}\frac{\Gamma(n/2+s/2)}{\Gamma(n/2)}, \tag{3.2}$$

where Γ denotes Euler's Gamma function. Then, it follows that we have three possible cases:

(i) If $\sigma = \bar{\sigma}_0(s,d)$ where $\bar{\sigma}_0(s,d) := \frac{1}{\sqrt[s]{I_0(s,d)}}$, then the network preserves the s-th moment of the layer outputs, i.e. for every $k \geq 1$, $\mathbb{E}\left[\|x^{(k)}\|^s\right] = \|x^{(0)}\|^s$, whereas for any $p > s$, $\mathbb{E}\|x^{(k)}\|^p \to \infty$ exponentially fast in k.

(ii) If $\sigma < \bar{\sigma}_0(s,d)$, then $\mathbb{E}\left[\|x^{(k)}\|^s\right] \to 0$ exponentially fast in k.

(iii) If $\sigma > \bar{\sigma}_0(s,d)$, then $\mathbb{E}\left[\|x^{(k)}\|^s\right] \to \infty$ exponentially fast in k.

Remark 2. ***($s = 2$ case)*** *In the special case of $s = 2$, Theorem 1 yields $I_0(2,d) = d/2$ and $\bar{\sigma}_0^2(2,d) = 2/d$ which corresponds to Kaiming initialization, details of this derivation is in Remark 16 in the supplementary material.*

Remark 3. ***(Variable width*** d_k***)*** *If the width d_k of layer k is not a constant equal to d but instead varying over k, then our analysis extends to this case naturally where it would suffice to replace the formula (3.1) with $\mathbb{E}\left[\|x^{(k)}\|^s\right] = \|x^{(0)}\|^s(\sigma^s\prod_{j=1}^k I_0(s,d_j))$.*

A natural question that arises is how does the critical variance depend on d and s when d is large. The next result gives precise asymptotics for $\bar{\sigma}_0(s,d)$ in the large d regime. The result relies on careful asymptotics for the Gamma functions and binomial coefficients arising in Theorem 1.

Corollary 4. ***(Critical variance*** $\bar{\sigma}_0(d,s)$ ***when*** d ***is large)*** *For fixed width d and $s \in (0,2]$, we have*

$$\bar{\sigma}_0^2(s,d) = \tfrac{2}{d} + \tfrac{5(2-s)}{2d^2} + o(\tfrac{1}{d^2}),$$
$$\bar{\sigma}_0(s,d) = \tfrac{\sqrt{2}}{\sqrt{d}} + \tfrac{5\sqrt{2}(2-s)}{8d\sqrt{d}} + o(\tfrac{1}{d\sqrt{d}}).$$

Therefore, it follows from Theorem 1 that if $\sigma^2 = \frac{2}{d} + \frac{5(2-s)}{2d^2}$, then the network will preserve the moment of order $s + o(\frac{1}{d})$ of the network output.

According to Corollary 4, $\log(\bar{\sigma}_0^2(s,d) - \frac{2}{d}) \approx -2\log(d) + \log(\frac{5(2-s)}{2})$ for large d. This is illustrated on the left panel of Figure 1 where we plot $\log(\bar{\sigma}_0^2(s,d))$ vs. $\log(d)$ based on the formula (3.1) where we observe the relationships is a straight line with slope approximately -2 as predicted by our theory. The right panel of Figure 1 illustrates part (*iii*) of Theorem 1 about how the moments can grow if we choose $\sigma = \bar{\sigma}_0(s,d)$ depending on the value of s.

It is not hard to show that under Gaussian initialization with ReLU activation, the network output can be zero with a non-zero probability (see Lemma 17 in the appendix), which is related to the known "dying neuron" problem associated with ReLU activations (Lu et al., 2019) about the fact that ReLU networks may output zero frequently. The choice of σ will clearly affect the variance of $x^{(k)}$ (see Theorem 1), however it won't affect the probability that the k-th layer output $x^{(k)} = 0$. A natural question that arises is what is the effect of σ on the growth rate of $\|x^{(k)}\|$ conditional on the event that $x^{(k)} \neq 0$. For this purpose, given an initial point $x^{(0)} \in \mathbb{R}^d$ fixed, we consider the conditional

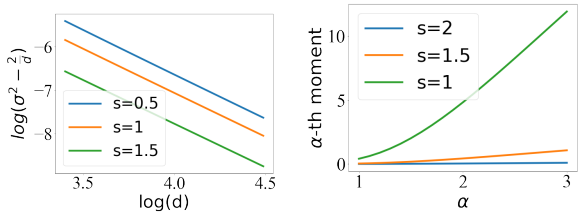


Figure 1: *Left*: Plot of $\log(\bar{\sigma}_0^2(s,d) - \frac{2}{d})$ vs. $\log(d)$ for ReLU. *Right*: Growth of the α-th moment of $x^{(k)}$ for $k = 500$ and $\sigma = \bar{\sigma}_0(s,d)$ with different s for $d = 64$. When s gets smaller, moments grow faster.

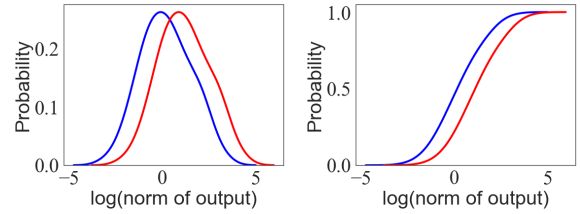


Figure 2: *Left:* Probability density $f_k(r)$ of $R_{k,0}$. *Right:* Cumulative density function (cdf) of $R_{k,0}$. The blue line in both figures is the result of Kaiming's method, which corresponds to $s = 2$. The red line in both figures is the result of our method with $s \approx 1$.

probability density function of $\log(\|x^{(k)}\|)$ given that $x^{(k)} \neq 0$, i.e.

$$f_k(r) := \mathbb{P}\left(\log(\|x^{(k)}\|) = dr \mid x^{(k)} \neq 0, a = 0\right). \quad (3.3)$$

Let $R_{k,0}$ be the random variable corresponding to the density $f_k(r)$. The quantity

$$\mu_0(\sigma) := \lim_{k\to\infty} \frac{R_{k,0}}{k} \quad (3.4)$$

is a measure of how fast the norm of the output of the layers of the network would grow if we would allow infinitely many layers. It is closely related to the *top Lyapunov exponent* in the probability and dynamical systems literature (Arnold et al., 1986; Cohen and Newman, 1984) which arises in the study of random Lipschitz maps, see e.g. (Elton, 1990). In the next result, we will obtain an explicit formula for $\mu_0(\sigma)$ (that depends on σ and dimension d), showing that $\mu_0(\sigma)$ is deterministic and does not depend on the initial point $x^{(0)}$. Furthermore, we show that a properly scaled $R_{k,0}$ converges to a Gaussian random variable in distribution, with an explicit mean and variance we can characterize.

Theorem 5. ***(Asymptotic normality of the log. of the norm of the network output)*** *Consider a fully connected network with an input $x^{(0)} \in \mathbb{R}^d$ and Gaussian initialization satisfying **(A1)-(A2)** with ReLU activation function $\phi_0(x) = \max(x, 0)$. Let $f_k(r)$ be the conditional probability density function of $\log(\|x^{(k)}\|)$ given that $x^{(k)} \neq 0$, defined formally by (3.3). Let $R_{k,0}$ be the random variable corresponding to the density $f_k(r)$. Then, the limit $\mu_0(\sigma)$ defined in (3.4) exists, it is deterministic and independent of $x^{(0)}$, satisfying the following formula:*

$$\mu_0(\sigma) = \log(\sigma) + \frac{1}{2}\sum_{n=1}^{d} \pi_d(n)\left[\log(2) + \psi_0\left(\frac{n}{2}\right)\right]. \quad (3.5)$$

Furthermore, $\frac{R_{k,0} - \mu_0(\sigma)k}{\sqrt{k}} \Rightarrow \mathcal{N}(0, s_0^2)$ in distribution as $k \to \infty$ with

$$s_0^2 = \frac{1}{4}\sum_{n=1}^{d} \pi_d(n)\left[\psi_1(\frac{n}{2}) + \left[\log(2) + \psi_0\left(\frac{n}{2}\right)\right]^2\right] - \frac{1}{4}\left(\sum_{n=1}^{d} \pi_d(n)\left[\log(2) + \psi_0\left(\frac{n}{2}\right)\right]\right)^2,$$

where ψ_0 is the di-gamma function, ψ_1 is the tri-gamma function and $\pi_d(n) = \binom{d}{n}\frac{1}{2^d - 1}$.

Remark 6. ***(First-order stochastic dominance property compared to Kaiming's method)*** *Theorem 5 shows that the logarithm of the norm of the k-th layer output $R_{k,0}$ will be asymptotically normal as $k \to \infty$ if $R_{k,0}$ is properly scaled, where the choice of σ will only affect the mean (but not the variance) of the asymptotic normal distribution. This is illustrated in Figure 2 where we plot the probability density function (pdf) on the left panel and the cumulative density function (cdf) of $R_{k,0}$ on the right panel where the pdf of $R_{k,0}$ has a Gaussian shape. We compare two initializations $\sigma^2 = \frac{2}{d}$ (Kaiming initialization which preserves variances) and our initialization technique $\sigma^2 = \frac{2}{d} + \frac{5}{2d^2}$ which preserves the moment of order $s = 1 + o(\frac{1}{d})$. We used $k = 100$ layers and dimension $d = 64$. We observe from the cdf's of network outputs on the right panel of Figure 2 that with our choice of σ, the norm of the network output is larger in the sense that it has first-order stochastic dominance (Hadar and Russell, 1969) relative to Kaiming initialization. Since our results also admit non-asymptotic versions (Remark 12), this dominance property will hold provably for large enough but finite k as well (due to the fact that our initialization results in a larger mean value $\mu_a(\sigma)$ in the setting of Theorem 1).*

4 Parametric ReLU, Randomized Leaky ReLU and Linear Activations

For parametric ReLU activations with $a > 0$, we develop an analogous result to Theorem 1 which charac-

terize s-th moments of the k-th layer output $x^{(k)}$ for $s \in (0, 2]$.

Theorem 7. ***(Explicit characterization of the critical variance*** $\bar{\sigma}_a^2(s,d)$***)*** *Consider a fully connected network with an input* $x^{(0)} \in \mathbb{R}^d$ *and Gaussian initialization satisfying* ***(A1)–(A2)*** *with activation function* $\phi_a(x)$ *for any choice of* $a \in (0, 1]$ *fixed. Then, for any* $s \in (0, 2]$*, the output of the* k*-th layer satisfies*

$$\mathbb{E}\left[\|x^{(k)}\|^s\right] = \|x^{(0)}\|^s (\sigma^s I_a(s,d))^k \tag{4.1}$$

with

$$I_a(s,d) = 2^{s/2} \frac{1}{\Gamma(1-s/2)} \sum_{n=0}^{d} \binom{d}{n} \frac{1}{2^d} \sum_{k=0}^{\infty} w_{k,n} \mathrm{B}(k+1-\frac{s}{2}, \frac{d}{2}+\frac{s}{2}) \tag{4.2}$$

with the convention that $I_a(2,d) = (1+a^2)\frac{d}{2}$*, where* $\mathrm{B}(\cdot,\cdot)$ *is the Beta function and*

$$w_{k,n} = \frac{1}{2}(1-a^2)^k \left[\binom{\frac{d-n}{2}+k-1}{k} n + a^2(d-n)\binom{\frac{d-n}{2}+k}{k}\right]. \tag{4.3}$$

Let $\bar{\sigma}_a(s,d) = \frac{1}{\sqrt[s]{I_a(s,d)}}$*. We have three possible cases:*

(i) If $\sigma = \bar{\sigma}_a(s,d)$*, then the network preserves the* s*-th moment of the layer outputs, i.e. for every* $k \geq 1$*,* $\mathbb{E}\left[\|x^{(k)}\|^s\right] = \|x^{(0)}\|^s$*, whereas for any* $p > s$*,* $\mathbb{E}\|x^{(k)}\|^p \to \infty$ *exponentially fast in* k*.*

(ii) If $\sigma < \bar{\sigma}_a(s,d)$*, then* $\mathbb{E}\left[\|x^{(k)}\|^s\right] \to 0$ *exponentially fast in* k*.*

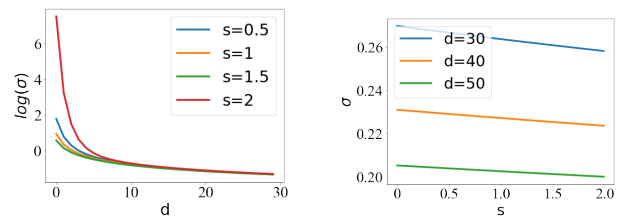
(iii) If $\sigma > \bar{\sigma}_a(s,d)$*, then* $\mathbb{E}\left[\|x^{(k)}\|^s\right] \to \infty$ *exponentially fast in* k*.*

Remark 8. ***(Extension to Randomized Leaky ReLU)*** *For Randomized Leaky ReLU activation,* a *is chosen randomly. Theorem 7 extends simply by replacing* $w_{k,n}$ *with* $\mathbb{E}[w_{k,n}]$ *where the expectation is taken with respect to the distribution of* a*. For instance, with a uniform distribution over an interval* $[\ell, u]$ *with default values of* $\ell = \frac{1}{3}$ *and* $u = \frac{1}{8}$ *(Xu et al., 2015),* $\mathbb{E}[w_{k,n}]$ *can be expressed with a closed-form formula as all the moments of the uniform distribution is explicitly known (Walck, 1996).*

We can also show that $\bar{\sigma}_a(s,d)$ possesses some monotonicity properties.

Corollary 9. ***(Monotonicity properties of*** $\bar{\sigma}_a(s,d)$***)*** *In the setting of Theorem 7, for* $(a,s,d) \in [0,1] \times (0,\infty) \times \mathbb{Z}_+$*, the function* $(a,s,d) \mapsto \bar{\sigma}_a(s,d)$ *is a monotonically (strictly) decreasing function of* a, d *and* s*.*

Figures 3a–3b illustrate $\bar{\sigma}_0(s,d)$ as a function of d when s is fixed where we see a monotonic behavior as proven in Corollary 9. We also observe in the figures that it is a monotonically decreasing function of s when d is fixed. Next, we characterize how $\bar{\sigma}_a(d,s)$ behaves for large d.



(a) $\log(\bar{\sigma}_0(s,d))$ versus d (b) $\bar{\sigma}_0(s,d)$ versus s

Figure 3: Dependency of $\bar{\sigma}_0(s,d)$ to parameters s and d.

Corollary 10. ***(Critical variance*** $\bar{\sigma}_a(d,s)$ ***when*** d ***is large)*** *For fixed width* d *and* $s \in (0, 2]$*, we have* $\bar{\sigma}_1^2(s,d) = \frac{1}{2}\left(\frac{\Gamma(\frac{d}{2})}{\Gamma(\frac{d}{2}+\frac{s}{2})}\right)^{2/s} = \frac{1}{d} + \frac{(2-s)}{2d^2} + o(\frac{1}{d^2})$ *with* $\bar{\sigma}_1^2(2,d) = \frac{1}{d}$ *in the special case* $s = 2$ *which corresponds to Lecun initialization. Therefore, it follows from Theorem 7 that if* $\sigma^2 = \frac{1}{d} + \frac{(2-s)}{2d^2}$*, then the network with linear activation will preserve the moment of order* $s + o(\frac{1}{d})$ *of the network output. More generally, for* $a > 0$ *small, we have*

$$\bar{\sigma}_a^2(s,d) = \left(\frac{2}{1+a^2}\right)\frac{1}{d} + \left(\frac{5-(12-\frac{5}{2}s)a^2}{2+(s+2)a^2}\right)\frac{(2-s)}{d^2} + \mathcal{O}(\frac{a^4}{d}) + o(\frac{1}{d^2}).$$

Similar to Corollary 4 for the ReLU case, we can express $\bar{\sigma}_a^2(s,d)$ as a function of s for large d. Thanks to Corollary 10, we can approximate $\bar{\sigma}_a^2(s,d)$ explicitly for Leaky ReLU with $a = 0.01$ without evaluating the double sums in (4.2). Leaky ReLU and linear activations do not output zero unless their input is zero; due to their piecewise linear structure. This is why, they can solve the "dying neuron" problem of ReLU activations to a certain extent (Lu et al., 2019). Consequently, under Gaussian initialization **(A1)**–**(A2)** with for Leaky ReLU and linear activations, i.e. when $a \in (0, 1]$, for any $\sigma > 0$ given, it is straightforward to show that $\mathbb{P}(x^{(k)} = 0) = 0$. Similar to our discussion for ReLU

activations, we introduce

$$f_{k,a}(r) := \mathbb{P}\left(\log(\|x^{(k)}\|) = dr \mid x^{(k)} \neq 0\right) = \mathbb{P}\left(\log(\|x^{(k)}\|) = dr\right),$$

where we used $\mathbb{P}(x^{(k)} = 0) = 0$ for $a \in (0, 1]$. Let $R_{k,a}$ be the random variable corresponding to the density $f_{k,a}(r)$. The quantity

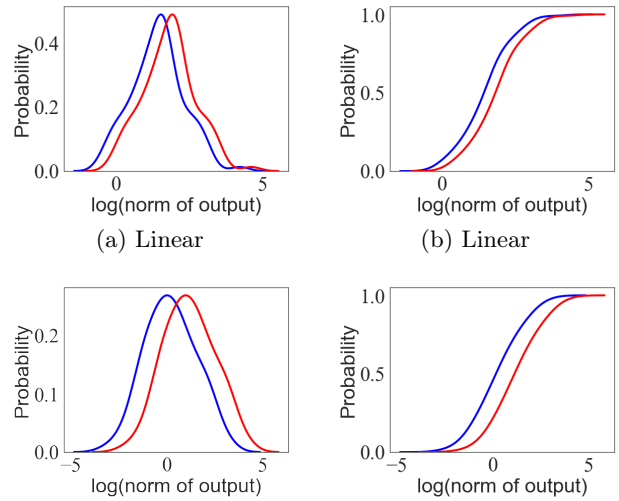
$$\mu_a(\sigma) := \lim_{k\to\infty} \frac{R_{k,a}}{k} \quad \text{for} \quad a \in (0, 1], \tag{4.4}$$

is called the *top Lyapunov exponent* for the random Lipschitz map $x^{k+1} = \phi_a(W^{k+1}x^k)$ where σ scales the W^{k+1} term. The following theorem derives an explicit formula for $\mu_a(\sigma)$ and shows that $R_{k,a}$ is asymptotically normal if it is properly scaled for parametric ReLU.

Theorem 11. ***(Asymptotic normality of the log. of the norm of the network output)*** *Consider a fully connected network with an input $x^{(0)} \in \mathbb{R}^d$ and Gaussian initialization satisfying **(A1)-(A2)** with Leaky ReLU activation function $\phi_a(x)$ with $a \in (0, 1]$. Let $f_k(r)$ be the conditional probability density function of $\log(\|x^{(k)}\|)$ given that $x^{(k)} \neq 0$, defined formally by (4). Let $R_{k,a}$ be the random variable corresponding to the density $f_{k,a}(r)$. Then, the limit $\mu_a(\sigma)$ defined in (4.4) exists, it is deterministic and independent of $x^{(0)}$, explicitly given by the formula (I.11) in the supplementary material. Let $R_{k,a}$ be the random variable corresponding to the density $f_{k,a}(r)$. Then, $\frac{R_{k,a}-\mu_a(\sigma)k}{\sqrt{k}} \Rightarrow \mathcal{N}(0, s_a^2)$ in distribution as $k \to \infty$ where s_a^2 is defined by (I.12) in the supplementary material.*

Remark 12. ***(Non-asymptotic version of Theorems 5 and 11 and stochastic dominance)*** *Theorems 5 and 11 are based on invoking the central limit theorem (CLT) in its proof. If we use a non-asymptotic version of the CLT instead such as the Berry–Esseen theorem (Berry, 1941), the results extend to finite k in a straightforward fashion. In Figure 4, we illustrate Theorem 11 where we plot the distribution of the natural logarithm of the norm of the output $R_{k,a}$ and observe a Gaussian behavior. In the supplementary material (Remark 18), we also discuss the stochastic dominance properties of $s < 2$ with respect to $s = 2$.*

Remark 13. ***(Extension of results to dropout)*** *Dropout is a popular technique that randomly removes some neurons to prevent overfitting (Srivastava et al., 2014). In this case, with zero bias, the layer recursion becomes $x^{(k+1)} := \phi_a(W^{(k+1)}(x^{(k)} \odot \varepsilon^{(k+1)})$ where $\odot$ denotes component-wise multiplication and $\varepsilon^{(k+1)}$ is a scaled Bernouilli random variable with i.i.d. components satisfying $\mathbb{P}(\varepsilon_i^{(k+1)} = 0) = 1 - q$ and $\mathbb{P}(\varepsilon_i^{(k+1)} = \frac{1}{q}) = q$ where q is the probability to keep a neuron with $q \in (0, 1]$ (see e.g. (Pretorius et al., 2018)). All the results in this paper generalize naturally with minor modifications (such as scaling with q) in the results if dropout is used (see Appendix J). For example, for any $s \in (0, 2]$ and $q \in (0, 1]$, the critical threshold for ReLU with dropout becomes $\sigma_{0,q}^2(s, d) = \frac{2q}{d} + \frac{2-s}{2d^2}(6 - q) + \mathcal{O}(\frac{1}{d^2})$ where our analysis recovers the results of Corollary 4 in the special case when $q = 1$ and results of Pretorius et al. (2018) when $s = 2$.*



(a) Linear (b) Linear (c) Leaky-ReLU (d) Leaky-ReLU

Figure 4: Distribution of the natural logarithm of the norm of the output $R_{k,a}$ through 100 layers with Linear ($a = 1$) and Leaky ReLU activation with $a = 0.01$. The blue line in all figures is the result of Kaiming's method, the red line is the result of our initialization. (a): Probability density of $R_{k,1}$ where choose $\sigma = \bar{\sigma}_1(1, d)$. (b): Cumulative density function of $R_{k,1}$. (c): Probability density of $R_{k,a}$ for $a = 0.01$ where we choose $\sigma = \bar{\sigma}_a(1, d)$. (d): Cumulative density function of $R_{k,a}$ for $a = 0.01$.

The following theorem shows that if bias vectors are zero, then with both Kaiming initialization and our initialization, the network outputs will converge to an almost sure limit of zero, even if the network outputs preserve moments of order s for every layer k. Roughly speaking, the reason this happens is that the network preserves the moments in a highly anisotropic manner, output is often zero but also can occassionally take large values so that s-th moment is preserved. This supports empirical results of (Saxe et al., 2013, Sec. 3) that observed this anisotropic behavior for linear activations. Our result shows that similar behavior happens with non-linear activations. We also show that depending on the sign of $\mu_a(\sigma)$, both L_p limit and a.s. limit can be only zero or infinity. In the

special case of $s = 2$ (for Kaiming initialization), such a convergence result in L_2 was previously proven in (Hanin and Rolnick, 2018, Thm. 5) where layer widths can take arbitrary values.

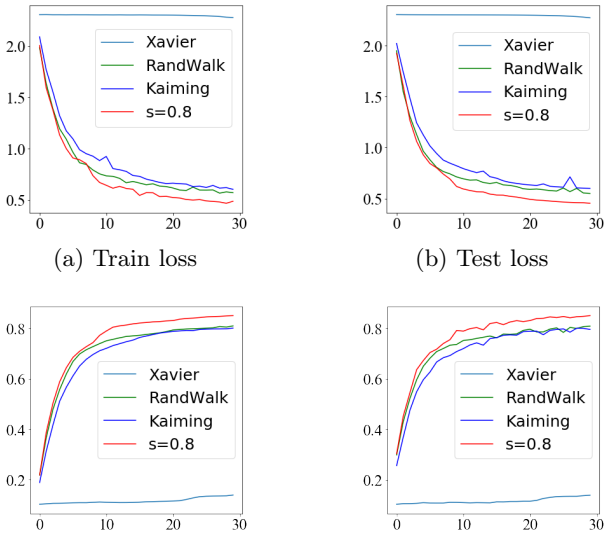
Theorem 14. ***(σ determines the almost sure (a.s.) and L_p limit)*** *Consider Gaussian initialization **(A1)**–**(A2)** with activation function $\phi_a(x)$ with $a \in [0, 1]$ and input $x^{(0)} \neq 0$. For ReLU, i.e. when $a = 0$, regardless of the choice of σ, the network output $x^{(k)}$ converges to zero a.s as $k \to \infty$. For parametric ReLU or for linear activations, i.e. when $a \in (0, 1]$, then $\mu_a(\sigma) < 0$ if and only if $\sigma = \bar{\sigma}_a(s, d)$ for some $s > 0$ and in this case $x^{(k)}$ converges to zero a.s. for $s \geq 1$ and for $s < 1$, $x^{(k)}$ converges in L_p for any $p \in (0, s)$ and has a subsequence that converges to zero almost surely, where $\mu_a(\sigma)$ is as in Theorem 11. On the other hand, if $\mu_a(\sigma) > 0$, then the sequence $x^{(k)}$ converges to infinity in L_p for any $p > 0$ and $x^{(k)}$ has a subsequence that converges to infinity a.s.*

If additive zero mean Gaussian noise is added to network outputs for linear activations, we can prove that the limit is non-zero and heavy tailed whereas the limit is zero without noise injection. Our results provides a theoretical support for experimental results of Poole et al. (2014) where additive noise was observed to improve performance by spreading information propagation more evenly across the network. The results also show that forward pass can make the network output (and subsequently the gradient of the training cost) heavy-tailed for Gaussian initialization.

Theorem 15. ***(Heavy-tailed a.s. limit)*** *Under Gaussian initialization **(A1)**–**(A2)** with a linear activation function $\phi_1(x)$, input $x^{(0)} \neq 0$ and $\sigma = \bar{\sigma}_1(s, d)$ for some $s \in (0, 2)$ where $\bar{\sigma}_1(s, d)$ is given explicitly in Corollary 10, if additive i.i.d. mean-zero Gaussian noise is added component-wise to post-activations, then the layer outputs $x^{(k)}$ admit a non-zero almost sure limit that is heavy tailed in the sense that it has infinite variance and its moments of order p are infinite for any $p > s$.*

5 Numerical Experiments

We compared our initialization method with Kaiming initialization (He et al., 2015) and Xavier method (Glorot and Bengio, 2010) on fully connected networks with linear, ReLU, Leaky ReLU activation functions. For the ReLU and linear activations, we also compared our method with random walk initialization (Sussillo and Abbott, 2014), which does not have explicit parameters for the Leaky ReLU but directly applicable to linear and ReLU activations. We only compare our method with initialization strategies that do not take additional CPU time during training for a fair comparison. We report train loss, test loss, train accuracy and test accuracy over first 30 epochs of training with SGD to focus on the impact of initialization on two benchmark problems: MNIST (LeCun et al., 1998a) and CIFAR-10 (Krizhevsky et al., 2009).



(a) Train loss (b) Test loss

(c) Train accuracy (d) Test accuracy

	train loss		test loss	
	mean	std	mean	std
Xavier	2.2761	0.0349	2.2723	0.0387
Randwalk	0.5712	0.4097	0.5498	0.4211
Kaiming	0.6039	0.426	0.6	0.42
s=0.8	**0.4877**	0.3059	**0.4535**	0.314
	train acc		test acc	
	mean(%)	std	mean(%)	std
Xavier	13.9	0.0299	13.99	0.0319
Randwalk	80.94	0.1649	80.82	0.1604
Kaiming	80.08	0.147	79.53	0.1488
s=0.8	**85.02**	0.1174	**84.98**	0.118

Figure 5: Fully connected network with width $d = 64$ and depth 20 for ReLU activation on MNIST. The plots are the average results over 20 runs, the mean and standard deviations (std) for runs are provided as a table. The x-axis represents the epoch number.

Figure 5 is the summary of our results for MNIST with ReLU activation with mean and standard deviation (std) of the runs reported over 20 runs, where we see a clear improvement with our initialization for $s = 0.8$. More specifically, within our initialization, we chose $\sigma^2 = \frac{2}{d} + \frac{3}{d^2}$ which preserves the moment $s \approx 0.8$ according to Corollary 4 where $d = 64$ over 20 layers. Further details of the experimental setup, results for ReLU and linear activations and our experiments on the CIFAR-10 dataset can be found in Section O of the appendix where we observed qualitatively similar results and our initialization method often improved

performance.

Heavy-tailed gradients at initialization. The works (Şimşekli et al., 2019b,a; Gürbüzbalaban et al., 2020) consider training of fully-connected and convolutional neural networks with SGD and standard initialization techniques and argue that the distribution of the gradients become often more and more heavy tailed over time. To be more specific, the numerical experiments in these works suggest that with traditional initialization approaches, the stochastic gradients have often light tails in the first epochs of SGD iterations but the tails become heavier over time as the number of epochs increases while the weights are being optimized. Such observations are also consistent and inline with the earlier results of Martin and Mahoney (2019). Also, these results together with Şimşekli et al. (2020) suggest that heavy tails often lead to better exploration and generalization properties. Our initialization technique allows this 'favorable heavy-tailed phase' to kick in earlier, right at the beginning of SGD iterations as opposed to later epochs of training. This is illustrated in Figure 6 which displays the tail index of gradient noise over iterations with our initialization, where the tail index is defined as the value of α such that the pdf $p(x)$ of the gradient noise is on the order of $1/\|x\|^{\alpha+1}$ when $\|x\|$ is large enough (see (Şimşekli et al., 2019b; Gürbüzbalaban et al., 2020) for more details on the tail index). Figure 6 is based on a fully-connected network with 5 layers with width 64. We use ReLU activation on the MNIST dataset where we take the batch size to be 32 with $s = 1$. We use the same estimator from Şimşekli et al. (2019b) for the tail index. We observe in Figure 6 the heavy tails arise starting from the initial iterations with a tail index α around 1 as expected.

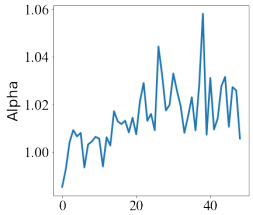
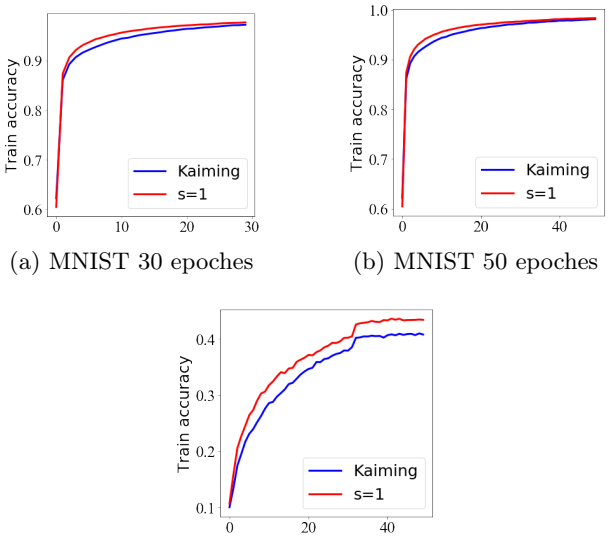


Figure 6: Tail index of gradient noise over epochs.

Results on convolutional neural networks. In Section N of the appendix, we provide the extensions of our theoretical results to convolutional neural networks. Here, we present our numerical experiments in Figure 7, where we used one convolutional layer and four fully-connected layers using ReLU with width $d = 64$ on MNIST and CIFAR-10 datasets. We train our networks with stochastic gradient (SGD). The stepsizes of both experiments are tuned and are same for the initializations. Figure 7a and 7b display the first 30 and 50 epoches of the training process of MNIST. With this architechture, after 50 iterations, we achieved an accuracy of %98.36 which is at a level of current state-of-the-art (see `https://benchmarks.ai/mnist` for benchmarks) on MNIST, where we see improvement compared to Kaiming initialization, especially in the first 30 epochs. Figure 7c shows the first 50 epochs of training processes on CIFAR-10, where we also see the improvement.



(a) MNIST 30 epoches (b) MNIST 50 epoches

(c) CIFAR-10

Figure 7: CNN on MNIST and CIFAR-10.

6 Conclusion

In this paper, we have developed a new class of initialization schemes for fully-connected neural networks with ReLU, parameteric ReLU, Leaky ReLU, Randomized Leaky ReLU or linear activations. Our schemes can preserve a fractional moment of order $s \in (0, 2]$ over the layer outputs therefore generalize existing schemes which correspond to the special case $s = 2$.

For all these schemes, we show that the network output admits a finite almost sure limit as the number of layers grows, and the limit is heavy-tailed in some settings. We also prove that the logarithm of the norm of the network outputs, if properly scaled, will converge to a Gaussian distribution with an explicit mean and variance we can compute. We also prove that our initialisation scheme avoids small network output values more frequently compared to traditional approaches, therefore can alleviate the dying neuron problem seen in ReLU networks that results in small network output values. We also provided numerical experiments that show that the new schemes can lead to improvement in the training process.

Acknowledgements

Mert Gürbüzbalaban and Yuanhan Hu acknowledge support from the grants NSF CCF-1814888, NSF DMS-2053485, NSF DMS-1723085 and Office of Naval Research Award Number N00014-21-1-2244.

References

Arnold, L., Kliemann, W., and Oeljeklaus, E. (1986). Lyapunov exponents of linear stochastic systems. In *Lyapunov Exponents*, pages 85–125. Springer.

Berry, A. C. (1941). The accuracy of the gaussian approximation to the sum of independent variates. *Transactions of the American Mathematical Society*, 49(1):122–136.

Blumenfeld, Y., Gilboa, D., and Soudry, D. (2019). A mean field theory of quantized deep networks: The quantization-depth trade-off. In Wallach, H., Larochelle, H., Beygelzimer, A., d Alché-Buc, F., Fox, E., and Garnett, R., editors, *Advances in Neural Information Processing Systems 32*, pages 7038–7048. Curran Associates, Inc.

Bottou, L. (1988). Reconnaissance de la parole par reseaux connexionnistes. In *Proceedings of Neuro Nimes 88*, pages 197–218, Nimes, France.

Bulmer, M. G. (1979). *Principles of Statistics*. Courier Corporation.

Buraczewski, D., Damek, E., Guivarc'h, Y., and Mentemeier, S. (2014). On multidimensional mandelbrot cascades. *Journal of Difference Equations and Applications*, 20(11):1523–1567.

Clevert, D.-A., Unterthiner, T., and Hochreiter, S. (2015). Fast and accurate deep network learning by exponential linear units (ELUs). *arXiv preprint arXiv:1511.07289*.

Cohen, J. E. and Newman, C. M. (1984). The stability of large random matrices and their products. *The Annals of Probability*, pages 283–310.

Cressie, N. and Borkent, M. (1986). The moment generating function has its moments. *Journal of Statistical Planning and Inference*, 13:337–344.

Daniely, A., Frostig, R., and Singer, Y. (2016). Toward deeper understanding of neural networks: The power of initialization and a dual view on expressivity. In *Advances In Neural Information Processing Systems*, pages 2253–2261.

Dauphin, Y. N. and Schoenholz, S. (2019). Metainit: Initializing learning by learning to initialize. In *Advances in Neural Information Processing Systems*, pages 12624–12636.

Defazio, A. and Bottou, L. (2019). Scaling laws for the principled design, initialization and preconditioning of ReLU networks. *arXiv preprint arXiv:1906.04267*.

Diaconis, P. and Freedman, D. (1999). Iterated random functions. *SIAM review*, 41(1):45–76.

Elton, J. H. (1990). A multiplicative ergodic theorem for lipschitz maps. *Stochastic Processes and their Applications*, 34(1):39 – 47.

Foss, S., Korshunov, D., Zachary, S., et al. (2011). *An Introduction to Heavy-tailed and Subexponential Distributions*, volume 6. Springer.

Fourdrinier, D., Strawderman, W. E., and Wells, M. T. (2018). *Spherically Symmetric Distributions*, pages 127–150. Springer International Publishing, Cham.

Glorot, X. and Bengio, Y. (2010). Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth International Conference on Artificial Intelligence and Statistics*, pages 249–256.

Gürbüzbalaban, M., Şimşekli, U., and Zhu, L. (2020). The heavy-tail phenomenon in sgd. *arXiv preprint arXiv:2006.04740*.

Hadar, J. and Russell, W. R. (1969). Rules for ordering uncertain prospects. *The American Economic Review*, 59(1):25–34.

Hanin, B. (2018). Which neural net architectures give rise to exploding and vanishing gradients? In *Advances in Neural Information Processing Systems*, pages 582–591.

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

Hanin, B. and Rolnick, D. (2018). How to start training: The effect of initialization and architecture. In *Advances in Neural Information Processing Systems*, pages 571–581.

Hayou, S., Doucet, A., and Rousseau, J. (2018). On the selection of initialization and activation function for deep neural networks. *arXiv preprint arXiv:1805.08266*.

He, K., Zhang, X., Ren, S., and Sun, J. (2015). Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1026–1034.

Ioffe, S. and Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*.

Kalman, B. L. and Kwasny, S. C. (1992). Why tanh: choosing a sigmoidal function. In *[Proceedings 1992]*

IJCNN International Joint Conference on Neural Networks, volume 4, pages 578–581. IEEE.

Klambauer, G., Unterthiner, T., Mayr, A., and Hochreiter, S. (2017). Self-normalizing neural networks. In *Advances in Neural Information Processing Systems*, pages 971–980.

Krizhevsky, A., Hinton, G., et al. (2009). Learning multiple layers of features from tiny images.

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

LeCun, Y., Bottou, L., Orr, G., and Muller, K.-R. (1998b). Efficient backprop. *Neural Networks: Tricks of the Trade. New York: Springer*.

Ling, Z. and Qiu, R. C. (2019). Spectrum concentration in deep residual learning: a free probability approach. *IEEE Access*, 7:105212–105223.

Lu, L., Shin, Y., Su, Y., and Karniadakis, G. E. (2019). Dying ReLU and initialization: Theory and numerical examples. *arXiv preprint arXiv:1903.06733*.

Luther, K. and Seung, H. S. (2019). Variance-preserving initialization schemes improve deep network training: But which variance is preserved? *arXiv preprint arXiv:1902.04942*.

Maas, A. L., Hannun, A. Y., and Ng, A. Y. (2013). Rectifier nonlinearities improve neural network acoustic models. In *In ICML Workshop on Deep Learning for Audio, Speech and Language Processing*.

Martin, C. H. and Mahoney, M. W. (2019). Traditional and heavy-tailed self regularization in neural network models. *arXiv preprint arXiv:1901.08276*.

Merkle, M. (1996). Logarithmic convexity and inequalities for the gamma function. *Journal of Mathematical Analysis and Applications*, 203(2):369–380.

Panigrahi, A., Somani, R., Goyal, N., and Netrapalli, P. (2019). Non-Gaussianity of stochastic gradient noise. *arXiv preprint arXiv:1910.09626*.

Poole, B., Sohl-Dickstein, J., and Ganguli, S. (2014). Analyzing noise in autoencoders and deep networks. *arXiv preprint arXiv:1406.1831*.

Pretorius, A., Van Biljon, E., Kroon, S., and Kamper, H. (2018). Critical initialisation for deep signal propagation in noisy rectifier neural networks. In *Advances in Neural Information Processing Systems*, pages 5717–5726.

Resnick, S. I. (2007). *Heavy-tail phenomena: Probabilistic and Statistical Modeling*. Springer Science & Business Media.

Saxe, A. M., McClelland, J. L., and Ganguli, S. (2013). Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. *arXiv preprint arXiv:1312.6120*.

Schoenholz, S. S., Gilmer, J., Ganguli, S., and Sohl-Dickstein, J. (2016). Deep information propagation. *arXiv preprint arXiv:1611.01232*.

Şimşekli, U., Gürbüzbalaban, M., Nguyen, T. H., Richard, G., and Sagun, L. (2019a). On the heavy-tailed theory of stochastic gradient descent for deep neural networks. *arXiv preprint arXiv:1912.00018*.

Şimşekli, U., Sagun, L., and Gürbüzbalaban, M. (2019b). A tail-index analysis of stochastic gradient noise in deep neural networks. In *International Conference on Machine Learning*, pages 5827–5837.

Şimşekli, U., Sener, O., Deligiannidis, G., and Erdogdu, M. A. (2020). Hausdorff dimension, stochastic differential equations, and generalization in neural networks. *arXiv preprint arXiv:2006.09313*.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958.

Steinsaltz, D. (1999). Locally contractive iterated function systems. *Annals of Probability*, pages 1952–1979.

Sussillo, D. and Abbott, L. (2014). Random walks: Training very deep nonlinear feed-forward networks with smart initialization. *CoRR, abs/1412.6558*, 287:300–302.

Sutskever, I., Martens, J., Dahl, G., and Hinton, G. (2013). On the importance of initialization and momentum in deep learning. In *International Conference on Machine Learning*, pages 1139–1147.

Tricomi, F. G., Erdélyi, A., et al. (1951). The asymptotic expansion of a ratio of Gamma functions. *Pacific Journal of Mathematics*, 1(1):133–142.

Walck, C. (1996). Hand-book on statistical distributions for experimentalists. Technical report, University of Stockholm Internal Report, SUF-PFY/96-01.

Xiao, L., Bahri, Y., Sohl-Dickstein, J., Schoenholz, S. S., and Pennington, J. (2018). Dynamical isometry and a mean field theory of cnns: How to train 10,000-layer vanilla convolutional neural networks. *arXiv preprint arXiv:1806.05393*.

Xu, B., Wang, N., Chen, T., and Li, M. (2015). Empirical evaluation of rectified activations in convolutional network. *arXiv preprint arXiv:1505.00853*.

Yang, G. and Schoenholz, S. (2017). Mean field residual networks: On the edge of chaos. In *Advances in Neural Information Processing Systems*, pages 7103–7114.

Zhang, J., Karimireddy, S. P., Veit, A., Kim, S., Reddi, S. J., Kumar, S., and Sra, S. (2019). Why ADAM beats SGD for attention models. *arXiv preprint arXiv:1912.03194*.