On the Importance of Full Rank Initializations in Deep Neural Networks

Anonymous Author(s)
Affiliation
Address
email

Abstract

Several methods have been proposed over the last few years for initializing the weights of neural networks in order to converge to better solutions or to reduce the time taken for convergence. On the other hand, there have been recent efforts connecting the full rank nature of weight matrices with the optimality of the final converged solution. In this work, we study the connection between popular initialization methods and the conditions necessary at optimal solution using deep linear networks with the squared loss. Through this connection, we attempt to provide a new explanation as to why these different initialization methods work well in practice.

1 Introduction

Over the past few years, several methods for initializing the weights of neural networks have been proposed. Although recent theoretical understanding claims that all local minima are perhaps equivalent [3], practical use of deep neural networks continues to rely on appropriate weight initialization methods. Glorot and Bengio [1] proposed a popular initialization method based on equalizing the variances of outputs of consecutive layers in a deep linear network. He *et al* [2] extended this work to provide a variant of this initialization. Few years later, Saxe *et al* [9] proposed their methods based on orthogonal weights. There have been continued efforts towards finding good initializations [7, 4, 10], due to their importance as suggested by Sutskever *et al* in [11]. On the other hand, there have been efforts connecting the full rank nature of weight matrices with the optimality of the final solution [12, 15, 13].

In this work, we seek to study the connection between well-known initialization methods that work well in practice and the necessary conditions imposed on the weight matrices at optimality. To the best of our knowledge, this is the first such study; although the findings are preliminary, we believe this is an interesting direction of study for the community to understand the training of deep neural networks.

2 Preliminaries

Consider a dataset of m points consisting of input-output pairs $(\mathbf{x}, \mathbf{y})$, where each $\mathbf{x}$ is of dimension d_x and each $\mathbf{y}$ is of dimension d_y, i.e., the inputs and outputs can be represented as matrices $X \in \mathbb{R}^{d_x \times m}$ and $Y \in \mathbb{R}^{d_y \times m}$ respectively. We consider a linear neural network of H hidden layers. Let the width of these H layers be $d_1, d_2, \ldots, d_H$. The width of the input and output layers are d_x and d_y respectively, which we call d_0 and d_{H+1} for consistency of notation. The weight matrices connecting the $(i-1)^{\text{th}}$ and the $(i)^{\text{th}}$ layers is denoted by $W_i \in \mathbb{R}^{d_i \times d_{i-1}}$ for $i = 1, 2, \ldots, H+1$. The output of this network can be written as a matrix product $W_{H+1} W_H \cdots W_1 X$, and our task is to minimize the sum-of-squares loss over the entire dataset defined below (note that our setup follows earlier efforts such as [3] and [12]):

$$L(W) \coloneqq \frac{1}{2} \left\| W_{H+1} W_H \cdots W_1 X - Y \right\|_F^2 \tag{1}$$

where W denotes the tuple of weight matrices $(W_{H+1}, W_H, \ldots, W_1)$ and $\|.\|_F$ denotes the Frobenius norm.

Let W_i^t represent the weight matrix W_i after $t(t > 0)$ iterations of gradient descent during training. We denote the initial weight matrix before gradient descent begins by W_i^0.

Assumptions: Our results follow [3] and [14], and hence use the same assumptions: $d_x \leq m$, $d_y \leq m$, and XX^T, YX^T are matrices of full rank. Furthermore, we assume that $d_0 \geq d_1 \geq \ldots \geq d_{H+1}$, i.e. the network is pyramidal in shape. Studying the results of this work relaxing these assumptions is an important direction of ongoing/future work.

3 Primary Claim

For convenience, we refer to commonly used initialization methods including Xavier initialization [1], He initialization [2], zero-mean (truncated) Gaussian initialization and Gaussian variants of Le Cun initialization [6] as 'well-known initialization methods' in this work.

Claim 1 (Informal)**.** *Well-known initialization methods converge to a global minimum for deep linear neural networks satisfying assumptions in Section 2 when trained using Gradient Descent (GD).*

Justification. Claim 1 will be shown as a result of the following sub-claims and results.

Sub-Claim 1. *Well-known initialization methods initialize the matrices to be full rank with high probability.*

Proof. The rank of a matrix of size $N \times n$ whose entries are sampled from a sub-Gaussian distribution with mean $\mathbf{0}$ is $\min\{N, n\}$ (i.e., the matrix is full rank) with high probability. The complete result can be found in [8]. Hence, initializing the matrix using zero-mean (truncated) Gaussian initialization and Gaussian variants of Le Cun initialization [6], Xavier initialization [1] and He initialization [2] result in full-rank initialization.

While these methods generate full-rank matrices with high probability, other popular methods such as orthogonal initialization [9] and identity initialization [5] are full-rank certainly, by construction. □

Lemma 1. *Consider two weight matrices W_i and W_{i+1}. If both of these matrices are full rank, then the product $W_{i+1}W_i$ is full rank.*

Proof. We use the rank of product and Sylvester's inequalities to prove this result. Note that $W_{i+1} \in \mathbb{R}^{d_{i+1} \times d_i}$ and $W_i \in \mathbb{R}^{d_i \times d_{i-1}}$. From our assumption on the width of the network, $d_{i+1} \leq d_i \leq d_{i-1}$, and hence $\mathrm{rank}(W_{i+1}) = d_{i+1}$ and $\mathrm{rank}(W_i) = d_i$, since both these matrices are full rank.

From the rank of product inequality, $\mathrm{rank}(W_{i+1}W_i) \leq \min\{d_{i+1}, d_i\} = d_{i+1}$. Using Sylvester's inequality, $\mathrm{rank}(W_{i+1}W_i) \geq d_{i+1} + d_i - d_i = d_{i+1}$. From the two inequalities, we get $\mathrm{rank}(W_{i+1}W_i) = d_{i+1}$. □

Corollary 1. *In well-known initialization methods, the product of initialized weight matrices is full-rank with high probability, for deep linear networks satisfying the assumptions in Section 2.*

Proof. This corollary directly follows from Lemma 1 and Sub-Claim 1. □

Using the corollary above, we conclude that $\mathrm{rank}(W_H^0 W_{H-1}^0 \ldots W_1^0) = d_y$.

Sub-Claim 2. *The product of rank of weight matrices, $\mathrm{rank}(W_H W_{H-1} \ldots W_1)$, follows a non-decreasing path over the iterations of Gradient Descent (GD). In other words, $\mathrm{rank}(W_H^{t+1} W_{H-1}^{t+1} \ldots W_1^{t+1}) \geq \mathrm{rank}(W_H^t W_{H-1}^t \ldots W_1^t)$ for all $t > 0$.*

Justification. We justify this claim empirically at this time, and conducted an exhaustive set of experiments with the assumptions considered earlier. We trained deep linear neural networks with variable number of layers (ranging from 2 to 5) and of different sizes (ranging from 500 to 50), and plotted the variation of the rank of the product of weight matrices with iterations while training with GD. Owing to space constraints, we present herewith (Figures 1 and 2) sample plots supporting our claim. Our comprehensive experiments suggested that these claim hold almost all the time under the aforementioned assumptions, and proving this theoretically is part of our ongoing/future efforts.

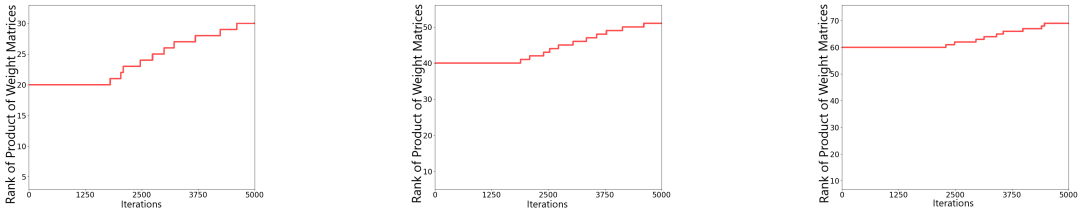

Figure 1: Left, middle, right sub-plots show results of Gradient Descent (GD) over iterations obtained on a `200 x 166 x 133 x 100` neural network architecture with initial rank of 20, 40 and 60 respectively for the product of weight matrices (highest rank possible: 100).

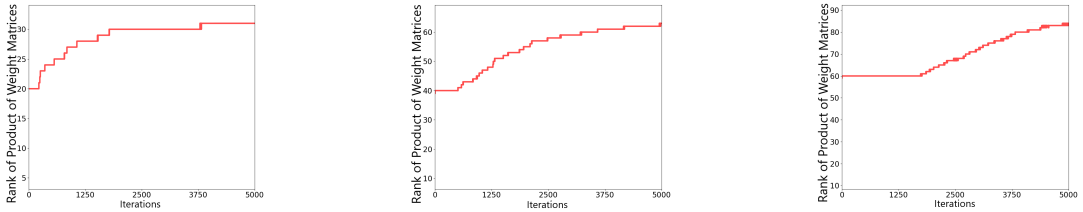

Figure 2: Left, middle, right sub-plots show results of Gradient Descent (GD) over iterations on a `500 x 437 x 375 x 312 x 250` neural network architecture obtained with initial rank of 20, 40 and 60 respectively for the product of weight matrices (highest rank possible: 250).

□

Lemma 2. *Under the assumptions and constructions stated so far,* $\text{rank}(W_H^t W_{H-1}^t \dots W_1^t) = d_y$ *for all* $t \geq 0$*, when* $\text{rank}(W_H^0 W_{H-1}^0 \dots W_1^0) = d_y$*.*

Proof. Combining the facts that $\text{rank}(W_H^0 W_{H-1}^0 \dots W_1^0) = d_y$, $\text{rank}(W_H^t W_{H-1}^t \dots W_1^t) \leq d_y$ for any t, and that $\text{rank}(W_H^{t+1} W_{H-1}^{t+1} \dots W_1^{t+1}) \geq \text{rank}(W_H^t W_{H-1}^t \dots W_1^t)$ for all $t \geq 0$, we get that $\text{rank}(W_H^t W_{H-1}^t \dots W_1^t) = d_y$. □

Continuing from Corollary 1 and Lemma 2, we infer that the product of the weight matrices remain full rank during training using GD with well-known initialization methods in these settings. To summarize, we hypothesize that the listed well-known methods initialize networks with weight matrices in order that the product of weight matrices is full-rank (not necessarily with this objective though) and since GD follows a non-decreasing path of rank of product of weight matrices, the product of weight matrices remains full-rank throughout training.

We now connect the above observations to a result from [14] which provides necessary and sufficient conditions to check whether a given optimization method has converged to a global optimum. Under certain conditions, the result classifies the set of critical points into global optima or saddles based on the rank of product of weight matrices as specified below:

Theorem 1 (Thm 2.1, [14])**.** *If* $k = \min\{d_x, d_y\}$*, consider* $\mathcal{V}_1 := \{(W_1, \dots, W_{H+1}) : \text{rank}(W_{H+1} W_H \dots W_1) = k\}$*. Every critical point of* $L(W)$ *in* $\mathcal{V}_1$ *is a global minimum and every critical point of* $L(W)$ *in* $\mathcal{V}_1^c$ *is a saddle point.*

The sub-claims above and Theorem 1 allow us to infer that when using GD to train deep linear networks with the above assumptions, the training converges to a critical point (i.e. model) whose product of weight matrices is full rank, and hence results in a global minimum.

Claim 2. *Any initialization that ensures that* $\text{rank}(W_H^0 W_{H-1}^0 \dots W_1^0) = d_y$ *in a linear network satisfying assumptions of Section 2 would lead to a convergence to a global minimum if GD converges.*

Proof. The proof is straightforward from Sub-claim 2 and Theorem 1. Sub-claim 2 and Lemma 2 together imply that $\text{rank}(W_H^t W_{H-1}^t \dots W_1^t) = d_y$ for $t \geq 0$. And so when GD converges, the product of weight matrix is full rank, which in conjunction with theorem 1 implies that GD would converge to global minima. □

The above claim gives a sufficient condition for any initialization to be good enough to lead to global convergence for deep linear networks.

We studied this hypothesis using a comprehensive set of experiments again, with a sample result shown in Figure 3. These empirical studies support our hypothesis that well-known initialization methods ensure that $\text{rank}(W_H^0 W_{H-1}^0 \dots W_1^0) = d_y$, although not explicitly derived keeping this condition in mind. Beyond the objectives considered for these methods, we surmise that the connection that leads to their empirical success is perhaps that they all satisfy the above-mentioned sufficient condition (Claim 2).

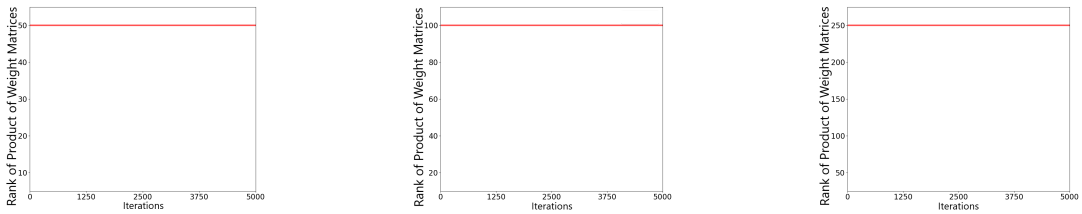

Figure 3: Variation of rank with iterations for three networks and different initialization schemes. *(Left:)* Network architecture `100 x 75 x 50` with Gaussian/normal variant of Xavier's initialization [1], *(Middle:)* Network architecture `200 x 175 x 150 x 125 x 100` with Gaussian/normal variant of He's initialization[2], *(Right:)* Network architecture `500 x 416 x 333 x 250` with Saxe's initialization [9]. Note that in all the cases, the initialization results in a full rank product and there is no decrease of rank over the iterations.

3.1 Some remarks on the claims

We now present some remarks to dispel misconceptions that might lead from our results. Firstly, we do not claim that GD does not converge to global minima if $\text{rank}(W_H^0 W_{H-1}^0 \dots W_1^0) < d_y$. We observe empirically that the rank may increase over iterations, such that $\text{rank}(W_H^t W_{H-1}^t \dots W_1^t) = d_y$ after a few iterations (which is in accordance with Sub-claim 2), which can result in global convergence. We observe this phenomenon in some of our experiments, presented below in Figure 4 (first two plots). In fact, it seems that when the initialization is such that $\text{rank}(W_H^0 W_{H-1}^0 \dots W_1^0) \approx d_y$, it is almost always the case that $\text{rank}(W_H^t W_{H-1}^t \dots W_1^t) = d_y$ after a finite number of iterations.

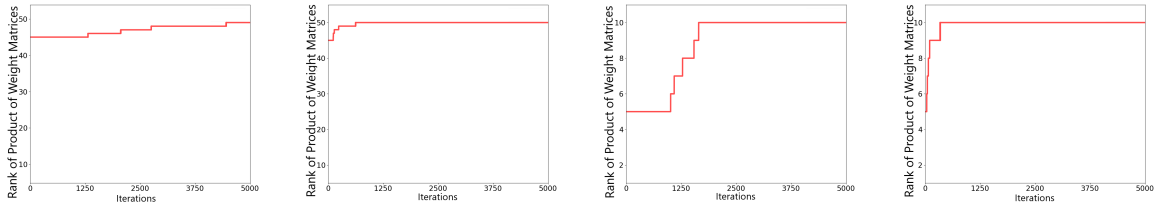

Figure 4: The first two plots show the variation of rank with iterations when starting with a rank close to full rank for architectures `100 x 75 x 50` and `100 x 83 x 66 x 50` (full rank: 50 in both scenarios), while the last two show the variation when starting with a low rank for the same architectures. In the first two plots, the rank becomes full i.e., 50, while it saturates at 10 (which is not full rank) in the last two plots.

We also add that although it might seem from the above experiments that if GD were run for a very large number of iterations, it will always lead to product of weight matrices being full rank; this however is not the case. In fact, if the initialization is such that $\text{rank}(W_H^0 W_{H-1}^0 \dots W_1^0)$ is not very close to d_y, the rank of the product of weight matrices, $\text{rank}(W_H^t W_{H-1}^t \dots W_1^t)$, saturates after a certain number of iterations without reaching full rank. The last two plots in Figure 4 show this phenomenon. It is clear from Theorem 1 that in this case as $\text{rank}(W_H^t W_{H-1}^t \dots W_1^t) \neq d_y$, the network has not converged to global minimum, but to a local minimum or a saddle point.

Conclusions and Future Work. In this work, we studied the connection between weight initialization methods and the rank of the product of the weight matrices, and showed interesting observations that could have impact on the training of deep neural networks. Our future work includes studying the aforementioned claims relaxing the assumptions made, as well as the theoretical justification of claims shown empirically in this work.

References

[1] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural network. 2010.

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

[3] Kenji Kawaguchi. Deep learning without poor local minima. In *Proceedings of Advances in Neural Information Processing Systems*, 2016.

[4] Günter Klambauer, Thomas Unterthiner, Andreas Mayr, and Sepp Hochreiter. Self-Normalizing Neural Networks. *arXiv e-prints*, page arXiv:1706.02515, Jun 2017.

[5] Quoc V. Le, Navdeep Jaitly, and Geoffrey E. Hinton. A Simple Way to Initialize Recurrent Networks of Rectified Linear Units. *arXiv e-prints*, page arXiv:1504.00941, Apr 2015.

[6] Yann A. LeCun, Léon Bottou, Genevieve B. Orr, and Klaus-Robert Müller. Efficient backprop. 1998.

[7] Dmytro Mishkin and Jiri Matas. All you need is a good init. *arXiv e-prints*, page arXiv:1511.06422, Nov 2015.

[8] M. Rudelson and R. Vershynin. Smallest singular value of a random rectangular matrix. *Communications on Pure and Applied Mathematics*, 2009.

[9] Andrew M. Saxe, James L. McClelland, and Surya Ganguli. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. 2014.

[10] David Sussillo and L. F. Abbott. Random Walk Initialization for Training Very Deep Feedforward Networks. *arXiv e-prints*, page arXiv:1412.6558, Dec 2014.

[11] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In Sanjoy Dasgupta and David McAllester, editors, *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, pages 1139–1147, Atlanta, Georgia, USA, 17–19 Jun 2013. PMLR.

[12] Chulhee Yun, Suvrit Sra, and Ali Jadbabaie. Global optimality conditions for deep neural networks. *International Conference on Learning Representations*, 2018.

[13] Chulhee Yun, Suvrit Sra, and Ali Jadbabaie. Small nonlinearities in activation functions create bad local minima in neural networks. *arXiv e-prints*, page arXiv:1802.03487, Feb 2018.

[14] Chulhee Yun, Suvrit Sra, and Ali Jadbabaie. Small nonlinearities in activation functions create bad local minima in neural networks. *arXiv preprint arXiv:1802.03487*, 2018.

[15] Yi Zhou and Yingbin Liang. Critical Points of Neural Networks: Analytical Forms and Landscape Properties. *arXiv e-prints*, page arXiv:1710.11205, Oct 2017.