The Cardinal Edge

Volume 1 | Issue 3 Article 14

September 2023

Effects of Weight Initialization methods on FFN's

Ida K. Karem
Jefferson Community and Technical College, mtg.card.play2@gmail.com

Effects of Weight Initialization methods on FFN's

Cover Page Footnote

Acknowledgments: I would like to thank Zimeng Lyu and Joshua Karns for answering questions I had about their study on Weight Initialization Effects on Neuroevolutionary neural networks. I would also like to thank Siddharth Kumar for answering questions about his study on weight initialization in deep neural networks.

Effects of Weight Initialization methods on FFN's

Ida K. Karem[1]
[1]Jefferson Community and Technical College

ABSTRACT

Weight initialization is the method of determining starting values of weights in a neural network. The way this method is done can have massive effects on the network[2, 3, 6, 9] and can halt training if not handled properly. On the other hand, if initialization is chosen tactfully it can improve training and accuracy greatly. The initialization method usually called Normalized Xavier will be referred to as Nox in this paper to avoid confusion with the Xavier initialization method. This study analyzes five methods of weight initialization(Nox, He, Xavier, Plutonian, and Self-Root), two of them being new to this study combined with three activation functions(Relu, Swish, and Tanh) and uses two datasets(MNIST[5], US Census 1990[4]). The study compares weight initialization methods using average MSE's of FFN's and shows significance by using MannWhitney U p-tests. This study does not provide very many definitive results outside of what is already proven in other studies but does provide a lot of new questions and speculation that can hopefully be answered. The definitive data this study does provide is as follows. While Swish is the activation function for all layers, the Plutonian produces lower error than the He, Nox, and Xavier, and the Xavier produces higher error than any other initialization method with statistical significance. The Self-Root produces higher error than any other initialization method while Tanh is the activation function for all layers. When Relu was the activation function in all layers Nox and He had a very significant statistical similarity. As for speculation, the Plutonian proved to be quite flexible in its use, possibly indicating a low error if used in networks with different activation functions in different layers. The Nox networks with Tanh as an activation performed better on MNIST, which could mean that when Tanh is an activation function more neurons per layer could lead to less error with the Nox.

KEYWORDS: neural networks, weight initialization, data science

INTRODUCTION

The effects of weight initialization on neural networks are extremely important. Bad weight initialization can lead to a myriad of problems such as exploding or vanishing gradients[2]. These problems can get worse depending on the type of activation[2, 3]. The point of this study is to test the effectiveness of old activation functions (Xavier, Nox, and He) as well as some new ones (Self-Root and Plutonian) and determine what works best for simple feed forward networks (FFNs) using a certain type of activation function. My intent in publishing this study is that researchers can take this knowledge and apply it to more complex networks such as CNNs and RNNs, for example.

INITIALIZATION

Xavier weight initialization is a method developed by Glorot and Bengio in 2010[2] The Nox is defined in [2], it was created by studying the variance of layers with linear activation functions but was found to work well with things like Tanh and Sigmoid. The He (sometimes called the Kaiming) weight initialization was developed by Kaiming, Xiangyu, Shaoqing, and Jian in 2015[3] to use with Relu. To quote the creators of He: "The main difference between our derivation and the Xavier initialization is that we address the rectifier nonlinearities" – page 5, [3], 2015

I developed the Self Root and the Plutonian to test the effect of initialization methods with comparatively steep slopes. I used a normal distribution for the Self Root so I could test if methods using that distribution had better network performances.

$$Xavier(n) = U\left[-n^{-\frac{1}{2}}, n^{-\frac{1}{2}}\right] \quad (1)$$

$$He(n) = G\left[0, \sqrt{\frac{2}{n}}\right] \quad (2)$$

$$Nox(n, m) = U\left[-\frac{\sqrt{6}}{\sqrt{n+m}}, \frac{\sqrt{6}}{\sqrt{n+m}}\right] \quad (3)$$

$$Self\ Root(n) = G\left[0, (n+20)^{\frac{4}{(n+20)}} - 1\right] \quad (4)$$

$$Plutonian(n) = U\left[-\frac{10n}{n^{1.85}}, \frac{10n}{n^{1.85}}\right] \quad (5)$$

distribution, and G[mean, standard dev] is a normal distribution. The values of the weights are determined by selecting a random number from the distribution

METHODOLOGY

Utilization This study utilizes the US Census (1990) dataset from UCI[4] to determine the sex of the people in the dataset. I will also be using the MNIST dataset[5] to determine the correct digit using the 28x28 grid from each handwritten image in the dataset. Each dataset was normalized. The values in the grid from MNIST had values ranging from 0-255 and were each

where n is the number of neurons from the previous layer, m is the number of neurons from the current layer. U[lower, higher] is a uniform divided by 255 to put them in a range from 0-1. The US census (1990) dataset was already normalized to one. Each dataset had 300 Networks trained on its data; these networks used each initialization method described previously combined with the activation functions Relu, Swish, and Tanh 20 times each.

Each network used the same activation function in all layers. All networks were trained on the first 25,000 points of data from their dataset and completed backpropagation after going through 100 points of data (250 epochs). The testing data for each network was taken from the 50,000th-60,000th points of data from each dataset. There was no optimizer used for this study and no cross fold validation, the reason for this is because we wanted to keep this study as simple as possible and eventually publish a follow up study with more complex networks. The loss function selected for this study MSE version used in this study is defined as:

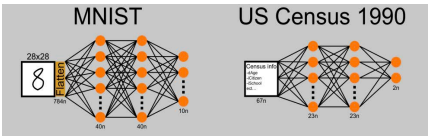
$$\sum_{i=1}^{n} (E_i - a_i)^2 \qquad (6)$$

Where E is the expected value, a is the value of the neurons in the last layer, and n is the number of neurons in the last layer. I used this version to make the MannWhitney U test more accurate. I will be evaluating the networks based on their average mean squared error(AMSE) from averaging all MSE's from the training data for each network.

The Methodology in this paper uses the methods described in a similar paper on the effects of weight initialization for Neuroevolutionary Networks[6]. Like the authors of [6] I will be taking the average MSE output from each network using the testing data (a total of 10000 MSE's were averaged for each of the 600 networks), putting them into a boxplot for comparison, and using MannWhitey U tests on the data to determine similarity. The MannWhitey U tests were conducted using the r language's wilcox.test function.

Architectures Every network uses the same Activation Function in each layer. All layers in all networks are fully connected. All networks use 0.045 as a learning rate because it's a value below 0.1 and not so low as to slow backpropagation. The framework used for this is the Rhymet Neural framework, you can find it here[7]. The networks use the following Architecture where n is the number of neurons.

I used 2 hidden layers because I thought it would be wise to make sure that a weight initialization that only preformed well with one hidden layer did not skew the results. I recommend more experiments be done to test if the results of this study still apply with deeper networks.



RESULTS

The results of the experiment, while not as conclusive as initially hoped, did provide a valuable amount of data leading to approval of and disapproval of some aspects of the hypothesis. The data collected also provided quite a few large outliers: as such, some of the data is located outside the boxplots. The full list of data can be found here[8].

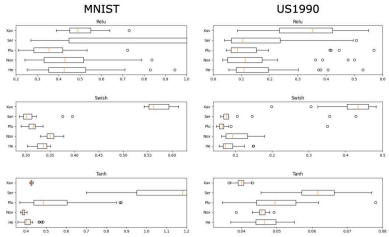


Figure 1: Box plots comparing the AMSE(Average Mean Squared Error) of all the networks using the initialization method (left of each plot) with all other networks that used the same activation function (top of each plot) and dataset (top of image). Plots made using Matplotlib

MNIST

1)

Relu	He	Nox	Plu	Ser	Xav
He	/	**0.698**	**0.052**	0.010	**0.086**
Nox	**0.698**	/	**0.114**	0.003	**0.072**
Plu	**0.052**	**0.114**	/	0.000	0.000
Ser	0.010	0.003	0.000	/	0.028
Xav	**0.086**	**0.072**	0.000	0.028	/

3)

Swish	He	Nox	Plu	Ser	Xav
He	/	0.000	0.000	0.001	0.000
Nox	0.000	/	0.000	0.000	0.000
Plu	0.000	0.000	/	0.046	0.000
Ser	0.001	0.000	0.046	/	0.000
Xav	0.000	0.000	0.000	0.000	/

5)

Tanh	He	Nox	Plu	Ser	Xav
He	/	0.003	0.000	0.000	0.005
Nox	0.003	/	0.000	0.000	0.000
Plu	0.000	0.000	/	0.000	0.000
Ser	0.000	0.000	0.000	/	0.000
Xav	0.005	0.000	0.000	0.000	/

US census 1990

2)

Relu	He	Nox	Plu	Ser	Xav
He	/	**0.512**	**0.369**	**1.000**	0.001
Nox	**0.512**	/	**0.947**	**0.620**	0.001
Plu	**0.369**	**0.947**	/	**0.478**	0.001
Ser	**1.000**	**0.620**	**0.478**	/	0.012
Xav	0.001	0.001	0.001	0.012	/

4)

Swish	He	Nox	Plu	Ser	Xav
He	/	0.040	0.004	**0.947**	0.000
Nox	0.040	/	0.000	0.046	0.000
Plu	0.004	0.000	/	0.002	0.000
Ser	**0.947**	0.046	0.002	/	0.000
Xav	0.000	0.000	0.000	0.000	/

6)

Tanh	He	Nox	Plu	Ser	Xav
He	/	**0.355**	**0.231**	0.000	0.000
Nox	**0.355**	/	0.043	0.000	0.000
Plu	**0.231**	0.043	/	0.000	0.000
Ser	0.000	0.000	0.000	/	0.000
Xav	0.000	0.000	0.000	0.000	/

Tables 1-6: MannWhitney U test p-values comparing the average MSE values of networks trained on one of two datasets(top) using a certain activation function(top left of each table). Statistically significant similarity is shown with bold. = 0.05

CONCLUSIONS

Hypothesis The hypothesis was i) He would outperform all of the other methods while using Swish and Relu and ii) Nox would outperform while using Tanh.

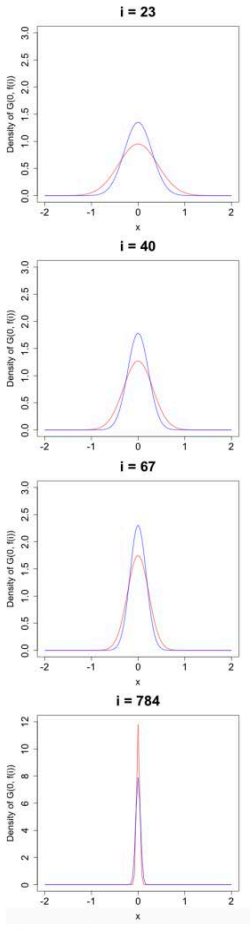
He He never had a median AMSE(Average Mean Squared Error) that was the highest or the lowest. While using the Relu activation function, He had a statistical similarity to the Nox and the Plutonian both times. He networks trained on the US1990 census dataset either had the second lowest median AMSE or had a statistical similarity to the initialization method with it. The median AMSE for Swish-He networks was lower than Swish-Nox and Swish-Xavier networks

with statistical significance on both occasions. The first component of my hypothesis was that He would outperform other networks while using Swish or Relu. It only had statistical similarities to the initialization method with the lowest median AMSE on both datasets using Relu so it did not statistically outperform it, only match it. When using Swish it was statistically outperformed by the Plutonian.

I find it surprising that the He network did not have the lowest error while paired with Relu as it is calculated as the optimal initialization method for Relu in [3, 9]. Relu-He FFN's only statistically outperformed the Relu-Xavier FFN's on MNIST. Other studies show that Relu-He outperforms Relu-Xavier (page 5 Figures 2 and 3, [3]), especially with deep neural networks. A possible explanation for Relu-He faltering here is the two hidden layer structure of the networks. As for the networks trained on Swish they had a higher median AMSE than other initialization methods with statistically significant difference. This is not very surprising as He was not designed with functions such as Tanh in mind[3]. Tanh-He networks did have lower error than some other initialization methods however it didn't have a similarity to the FFN's with the lowest error on either network.

Xavier The Xavier method consistently produced very high AMSE medians when paired with Relu or Swish. In the US Census Relu network, it was the only initialization method that was not statistically similar to any other method. On both of the Xavier-Swish networks, it had a minimum that was greater than all other method's maximum. While using Tanh, it had much lower results than prior and had the lowest median on the US 1990 Census network. For networks using Relu or Swish, I would not recommend Xavier, because most other initialization methods in this study outperformed it and it has been shown that it is not the optimal solution for these activation functions[3, 9]. The Xavier is the optimal initialization method for the Tanh activation according to [9] and Tanh-Xavier networks did have low error. If you are using a Tanh network it would seem that either the Nox or the Xavier is better for performance than other methods, but there are many unknowns that present themselves and as such this study can not be conclusive.

Self-Root The Self-Root method had the most variance between datasets and activations. While using the Relu or Tanh it had a high median AMSE, and a comparatively wide range of AMSE values for both datasets, especially so for the networks using MNIST. Using it with Swish, Self Root had very low median AMSE values but it did not have the lowest AMSE on either dataset. On the Swish and Relu networks using US 1990 Census as the dataset, the Self-Root and He had extremely high p-values (Tables 2 and 4), meaning that they have high statistically significant similarities. The networks using Tanh or the MNIST dataset had very low p-values for tests between He and Self-Root. It should be acknowledged that it did have some extreme outliers in its average AMSE data, and it did output NaN once. Because you cannot do a Mann-Whitney U test with a NaN value, that network was redone. The Self-Root method seems to be wildly inconsistent. Because its networks had high p values when tested with He's twice, I wanted to look further into measuring them. To compare the functions I made Figures 2-5(use this link to see an animation) with the i values being the inputs given when initialization is done for both network architectures (US census 2-3, MNIST 4-5). In all of the figures you can see that the distance between standard deviations is similar, however the probability density of one function is always higher.



Figures 2-5 Graphs of the functions He(Blue) and Self-Root(Red) with an input of i

I suspect the reason the MNIST networks are not statistically similar is because the probability density of the US 1990 networks are much closer than the probability densities of the MNIST networks. I do not know why the Tanh-He networks were not similar to the Self-Root networks. Swish-Root networks had low error but were never statistically similar to the Plutonian. This method is not similar to the optimal method you could get from using the equation in [9] (which is (0. 25)). I believe −0.5 more testing is necessary with this initialization method so we might have a better understanding of why it acts the way it does, and test the validity of the statements I have made in this section.

Nox The Nox method Had a lower median AMSE than the Xavier method with no statistical similarity four out six times.

It had a statistically significant similarity with the Plutonian and the He method while using the Relu. When Swish was the activation, it had the second highest median AMSE. When Tanh was the activation, it had a low median AMSE. Nox had the lowest median AMSE once and never had the highest AMSE.

The second component of my hypothesis was that Nox would outperform when Tanh was the activation. From the MNIST networks Nox did have the lowest median AMSE without any significant similarities, however the same could be said for the Xavier. My hypothesis was incorrect because Xavier outperformed Nox once.

On both datasets it shared statistical similarities with the He-Relu networks. This might suggest that Nox performs better with activation functions differentiable at 0. Having a lower median than Xavier-Tanh FFN's on MNIST could hint that Nox networks with more neurons in their hidden Tanh layers perform better, as in [2] when they used hidden layers with 1000 neurons in Tanh-Nox networks they had significantly lower test error percentages (Table 1, and Figures 11 and 12 on [2]) on 4 datasets. It should be noted that one of those datasets was also MNIST and they were using a different error function than I am. I recommend that more testing be done with Nox so we could have a better understanding of how activations non-differentiable at 0 and number of neurons in the hidden layers effect output. Both of the Swish-Nox networks underperformed compared to the other initialization methods, in contrast to the Relu-Nox. The networks using Tanh-Nox did seem to perform better and even outperformed Xavier once, I am unsure if there is enough data to conclude if Xavier is better than Nox or vice versa.

Plutonian The Plutonian had the lowest median AMSE while paired with the Relu. When using the Relu it had a significant similarity with He and Nox for both datasets. When Swish was the activation it had the second lowest median for MNIST and the lowest for the US 1990 Census. It had similar medians to the Self-Root on Swish, but the similarities were not statistically significant. On both the Tanh networks it had the second highest median, a low minimum, and a wide range of AMSE values.

The Relu-Plutonian networks had very low AMSE, but not enough to significantly distance itself from the He and Nox. I am unsure why Relu-Plutonian networks had a lower median than the Relu-He networks as He is the optimal method for Relu networks[9]. While using Swish it had no similarity to He and lower error meaning it's better than He for Swish networks. Because of its good performance on all three activation functions, It might have low error for networks using multiple activation functions. This would require testing but if true might shed some light on initialization for networks with multiple activations. Because of the versatility shown by the Plutonian in this study I would also suggest testing by applying it to the starting values of a convolution.

REFERENCES

1. Brownlee, J. (2021, February 7). *Weight initialization for deep learning neural networks.* Weight Initialization for Deep Learning Neural Networks. Retrieved July 16, 2022, from https://machinelearningmastery.com/weight-initialization-for-deep-learning-neural-networks/

2. Glorot, X., & Bengio, Y. (2010, March 31). *Understanding the difficulty of training deep feedforward neural networks.* Understanding the difficulty of training deep feedforward neural networks. Retrieved July 24, 2022, from https://proceedings.mlr.press/v9/glorot10a/glorot10a.pd

3. He, K., Zhang, X., Ren, S., & Sun, J. (2015, February 6). *ArXiv:1502.01852v1* [CS.CV] 6 feb 2015. Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. Retrieved July 24, 2022, from https://arxiv.org/pdf/1502.01852.pdf

4. Meek, C., Thiesson, B., and Heckerman, D. (1990). UCI Machine Learning Repository [https://archive.ics.uci.edu/ml/datasets/US+Census+Data+(1990)]. Irvine, CA: University of California, School of Information and Computer Science.

5. LeCun, Y., Cortes, C., & Burges, C. (2010, June 28). *The MNIST database.* MNIST handwritten digit database, from http://yann.lecun.com/exdb/mnist/

6. Lyu, Z., ElSaid, A. E. R., Karns, J., Mkaouer, M., & Desell, T. (2020, September 21). *ArXiv.org e-print archive.* An Experimental Study of Weight Initialization and Weight Inheritance Effects on Neuroevolution. Retrieved July 26, 2022, from https://arxiv.org/pdf/2009.09644.pdf

7. Karem, I. (2022, June 9). *github.com*. RhymetNeural. Retrieved March 9, 2023, from https://github.com/z8RtksLil4/RhymetNeural

8. Karem, I. (2022, December 9). *github.com*. StudyDataEWIFFN. Retrieved March 9, 2023, from https://github.com/z8RtksLil4/StudyDataEWIFFN

9. Kumar, S. (2017, May 4). *ArXiv.org e-print archive.* On weight initialization in deep neural networks. Retrieved March 9, 2023, from https://arxiv.org/pdf/1704.08863.pdf

Acknowledgments: I would like to thank Zimeng Lyu and Joshua Karns for answering questions I had about their study on Weight Initialization Effects on Neuroevolutionary neural networks. I would also like to thank Siddharth Kumar for answering questions about his study on weight initialization in deep neural networks.