ROBUST WEIGHT INITIALIZATION FOR TANH NEURAL NETWORKS WITH FIXED POINT ANALYSIS

Anonymous authors
Paper under double-blind review

ABSTRACT

As a neural network's depth increases, it can achieve strong generalization performance. Training, however, becomes challenging due to gradient issues. Theoretical research and various methods have been introduced to address this issues. However, research on weight initialization methods that can be effectively applied to tanh neural networks of varying sizes still needs to be completed. This paper presents a novel weight initialization method for Feedforward Neural Networks with tanh activation function. Based on an analysis of the fixed points of the function $\tanh(ax)$, our proposed method aims to determine values of a that prevent the saturation of activations. A series of experiments on various classification datasets demonstrate that the proposed method is more robust to network size variations than the existing method. Furthermore, when applied to Physics-Informed Neural Networks, the method exhibits faster convergence and robustness to variations of the network size compared to Xavier initialization in problems of Partial Differential Equations.

1 INTRODUCTION

Deep learning has enabled substantial advancements in state-of-the-art performance across various domains (LeCun et al., 2015; He et al., 2016). In general, the expressivity of neural networks exponentially increases with depth (Poole et al., 2016; Raghu et al., 2017), enabling strong generalization performance. This increased depth, though, can result in vanishing or exploding gradients and poor signal propagation throughout the model (Bengio et al., 1993), prompting the development of various weight initialization methods. Xavier initialization (Glorot & Bengio, 2010) ensures signals stay in the non-saturated region for sigmoid and hyperbolic tangent activations, while He initialization (He et al., 2015) maintains stable variance for ReLU networks. Especially in ReLU neural networks, several weight initialization methods have been proposed to mitigate the dying ReLU problem, which hinders signal propagation in deep networks (Lu et al., 2019; Lee et al., 2024). However, to the best of our knowledge, research on the initialization method to tackle the stability of extremely deep tanh networks during training is still limited. Such networks commonly use Xavier initialization (Raissi et al., 2019; Jagtap et al., 2022; Rathore et al., 2024) and are widely applied in various domains, such as Physics-Informed Neural Networks (PINNs) (Raissi et al., 2019) and Recurrent Neural Networks (RNNs) (Rumelhart et al., 1986), with performance often dependent on model size and initialization randomness (Liu et al., 2022).

The main contribution of this paper is the proposal of a simple weight initialization method for Feed-Forward Neural Networks (FFNNs) with tanh activation function. This method facilitates effective learning across a range of network sizes, outperforming Xavier initialization by reducing the need for extensive hyperparameter tuning such as the number of hidden layers and units. The theoretical foundation for this approach is provided through the fixed point of the function $\tanh(ax)$. We experimentally demonstrate that the proposed method achieves higher validation accuracy and lower validation loss compared to the Xavier initialization method across various FFNN network sizes on the MNIST, Fashion MNIST, and CIFAR-10 datasets. Additionally, the proposed method demonstrates its effectiveness in training across various network configurations within PINNs. Notably, while Xavier initialization shows decreasing loss as network depth increases, it fails to maintain performance beyond a certain depth, leading to increased loss and poor training outcomes. In contrast, the proposed method continues to improve performance even at greater depths, ensuring stable training and better results.

Contributions. Our contributions can be summarised as follows:

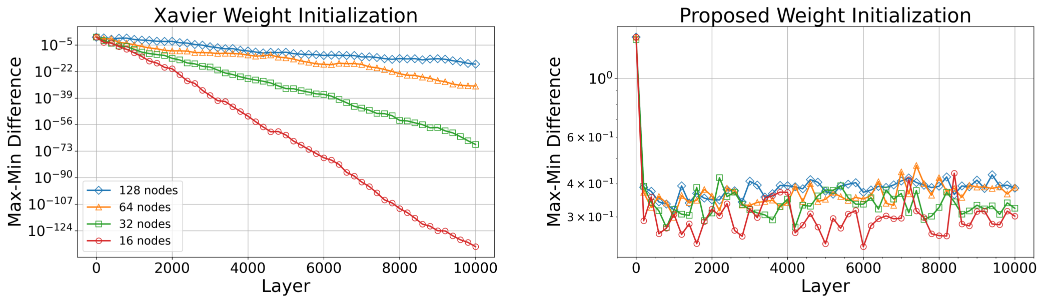
- We show the conditions under which activation values do not vanish as we increase the depth of the neural network, using a fixed-point analysis (Section 3.1 and 3.2).
- We propose a novel weight initialization method for tanh-based neural networks that has strong robustness to variations in network size (Section 3.2 and 3.3).
- We experimentally demonstrate that the proposed method is more robust to variations in network size than Xavier initialization on image benchmark datasets and PINNs (Section 4).

2 Related works

The expressivity of neural networks typically grows exponentially with depth, resulting in improved generalization performance (Poole et al., 2016; Raghu et al., 2017). Weight initialization is crucial for training deep networks effectively (Saxe et al., 2014; Mishkin & Matas, 2016). Xavier (Glorot & Bengio, 2010) and He He et al. (2015) initialization are common initialization methods typically used with tanh and ReLU activation functions, respectively. Various initialization methods have been proposed to facilitate the training of deeper ReLU neural networks (Lu et al., 2019; Bachlechner et al., 2021; Zhao et al., 2022; Lee et al., 2024). However, to the best of our knowledge, research on weight initialization for neural networks with tanh activation remains limited. Tanh neural networks have been increasingly used, particularly in physics-informed neural networks (PINNs).

PINNs have shown promising results in solving forward, inverse, and multiphysics problems arising in science and engineering. (Lu et al., 2021; Karniadakis et al., 2021; Cuomo et al., 2022b;a; Yin et al., 2021; Wu et al., 2023; Hanna et al., 2022; Bararnia & Esmaeilpour, 2022; Shukla et al., 2020; Zhu et al., 2024; Hosseini et al., 2023; Mao et al., 2020). PINNs approximate solutions to partial differential equations (PDEs) using neural networks and are typically trained by minimizing a loss defined by the sum of least-squares that incorporates the residual of PDE, boundary conditions, and initial conditions. This loss is usually minimized using gradient-based optimizers such as Adam (Kingma, 2014), L-BFGS (Liu & Nocedal, 1989), or a combination of both. Universal approximation theories (Cybenko, 1989; Hornik et al., 1989; Hornik, 1991; Park et al., 2020; Guliyev & Ismailov, 2018b; Shen et al., 2022; Guliyev & Ismailov, 2018a; Maiorov & Pinkus, 1999; Yarotsky, 2017; Gripenberg, 2003) guarantee the capability and performance of neural networks as an approximation of the analytic solution to PDE. However, PINNs still face challenges in accuracy, stability, computational complexity, and tuning optimal hyperparameters of loss terms. To alleviate these issues, many authors have introduced enhanced versions of PINNs: (1) the self-adaptive loss balanced PINNs (lbPINNs) that automatically adjust the hyperparameters of loss terms during the training process (Xiang et al., 2022), (2) the Bayesian PINNs (B-PINNs) that are specialized to deal with forward and inverse nonlinear problems with noisy data (Yang et al., 2021), (3) Rectified PINNs (RPINNs) that are trained with the gradient information from the numerical solution by the multigrid method and designed for solving stationary PDEs (Peng et al., 2022), (4) Auxiliary Pinns (A-PINNs) that effectively handle integro-differential equations (Yuan et al., 2022), (5) conservative PINNs (cPINNs) and exetended PINNs (XPINNs) that adopt the domain decomposition technique (Jagtap et al., 2020; Jagtap & Karniadakis, 2020), (6) parrel PINNs that reduces the computational cost of cPINNs and XPINNs (Shukla et al., 2021), (7) gradient-enhanced PINNs (gPINNs) that use the gradient of the PDE loss term with respect to the network inputs (Yu et al., 2022).

PINNs primarily employ Xavier initialization for training (Jin et al., 2021; Son et al., 2023; Yao et al., 2023; Gnanasambandam et al., 2023; Song et al., 2024), but our experimental results indicate that this method limits the performance of larger network sizes. Although there have been recent results on initialization methods for PINNs, most of them have relied on transfer learning (Tarbiyati & Nemati Saray, 2023). Thus, we propose a weight initialization method that does not require transfer learning and is robust to variations in network size.



(a) Xavier Initialization (b) Proposed Initialization

Figure 1: Difference between maximum and minimum activation values at each layer when propagating $3,000$ input data through a $10,000$-layer tanh FFNN, using Xavier initialization (Left) and the proposed initialization (Right). Experiments were conducted on networks with 10,000 hidden layers, each having the same number of nodes: 16, 32, 64, or 128.

3 Proposed Weight Initialization method

In this section, we discuss the proposed weight initialization method. Subsection 3.1 introduces the theoretical motivation behind the methodology. Subsection 3.2 presents how to derive the initial weight matrix that satisfies the conditions outlined in Subsection 3.1. Finally, in Subsection 3.3, we suggest the optimal hyperparameter σ_z in the proposed method.

3.1 Theoretical motivation

Experimental results in Figure 1 reveal that when Xavier initialization is employed in FFNNs with tanh activation, the distribution of activation values tends to cluster around zero in deeper layers. This vanishing of activation values can hinder the training process due to a discrepancy between the activation values and the desired output. However, theoretically preventing this phenomena is not straightforward. In this subsection, we gives a theoretical analysis based on a fixed point of $\tanh(ax)$ to bypass the phenomena. Before giving the theoretical foundations, consider the basic results for a tanh activation function. Recall that x^* is a fixed point of a function f if x^* belongs to both the domain and the codomain of f, and $f(x^*) = x^*$. The proofs of Lemma 1 and Lemma 2 are provided in Appendix A.

Lemma 1. *For a fixed $a > 0$ define the function $\phi_a : \mathbb{R} \to \mathbb{R}$ given as*

$$\phi_a(x) := \tanh(ax).$$

Then, there exists a fixed point x^*. Furthermore,

(1) *if $0 < a \leq 1$, then ϕ has a unique fixed point $x^* = 0$.*

(2) *if $a > 1$, then ϕ has three distinct fixed points: $x^* = -\xi_a$, 0, ξ_a such that $\xi_a > 0$.*

Remark that the function ϕ_a can be considered as one-layer tanh FFNN.

Lemma 2. *For a given initial value $x_0 > 0$ define*

$$x_{n+1} = \phi_a(x_n), \quad n = 0, 1, 2, \ldots.$$

Then $\{x_n\}_{n=1}^{\infty}$ converges regardless of the positive initial value $x_0 > 0$. Moreover,

(1) *if $0 < a \leq 1$, then $x_n \to 0$ as $n \to \infty$.*

(2) *if $a > 1$, then $x_n \to \xi_a$ as $n \to \infty$.*

Note that the parameter a in Lemma 2 does not change across all iterations. In Propositions 3 and Corollary 4, we address cases where the value of a varies with each iteration.

Proposition 3. *Let $\{a_n\}_{n=1}^{\infty}$ be a positive real sequence, i.e., $a_n > 0$ for all $n \in \mathbb{N}$, such that only finitely many elements are greater than 1. Suppose that $\{\Phi_m\}_{m=1}^{\infty}$ is a sequence of functions defined as for each $m \in \mathbb{N}$*

$$\Phi_m = \phi_{a_m} \circ \phi_{a_{m-1}} \circ \cdots \circ \phi_{a_1}.$$

Then for any $x \in \mathbb{R}$

$$\lim_{m\to\infty} \Phi_m(x) = 0.$$

Proof. Set $N = \max\{n | a_n > 1\}$. Define the sequences $\{b_n\}_{n=1}^{\infty}$ and $\{c_n\}_{n=1}^{\infty}$ such that $b_n = c_n = a_n$ for $n \leq N$, with $b_n = 0$ and $c_n = 1$ for $n > N$. Suppose that $\{\hat{\Phi}_m\}_{m=1}^{\infty}$ and $\{\tilde{\Phi}_m\}_{m=1}^{\infty}$ are sequences of functions defined as for each $m \in \mathbb{N}$

$$\hat{\Phi}_m = \phi_{b_m} \circ \phi_{b_{m-1}} \circ \cdots \circ \phi_{b_1}, \quad \tilde{\Phi}_m = \phi_{c_m} \circ \phi_{c_{m-1}} \circ \cdots \circ \phi_{c_1}.$$

Then, the inequality $\hat{\Phi}_m \leq \Phi_m \leq \tilde{\Phi}_m$ holds for all m. By Lemma 1, for any $x \geq 0$, we have $\lim_{m\to\infty} \hat{\Phi}_m = 0$ and $\lim_{m\to\infty} \tilde{\Phi}_m = 0$. Therefore, the Squeeze Theorem guarantees that $\lim_{m\to\infty} \Phi_m(x) = 0$.

□

Corollary 4. *Let $\epsilon > 0$ be given. Suppose that $\{a_n\}_{n=1}^{\infty}$ be a positive real sequence such that only finitely many elements are lower than $1 + \epsilon$. Then for any $x \in \mathbb{R} \setminus \{0\}$*

$$\left|\lim_{m\to\infty} \Phi_m(x)\right| \geq \xi_{1+\epsilon}$$

Proof. Set $N = \max\{n \mid a_n < 1 + \epsilon\}$. Define the sequence $\{b_n\}_{n=1}^{\infty}$ such that $b_n = a_n$ for $n \leq N$, and $b_n = 1 + \epsilon$ for $n > N$. The remainder of the proof is analogous to the proof of Proposition 3. □

3.2 The derivation of the proposed weight initialization method

To establish the notation, consider a feedforward neural network with L layers. The network processes K training samples, denoted as pairs $\{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^{K}$, where $\boldsymbol{x}_i \in \mathbb{R}^{N_x}$ is training input and $\boldsymbol{y}_i \in \mathbb{R}^{N_y}$ is its corresponding output. The iterative computation at each layer ℓ is defined as follows:

$$\boldsymbol{x}^\ell = \tanh(\mathbf{W}^\ell \boldsymbol{x}^{\ell-1} + \mathbf{b}^\ell) \in \mathbb{R}^{N_\ell} \quad \text{for all } \ell = 1, \ldots, L,$$

where $\mathbf{W}^\ell \in \mathbb{R}^{N_\ell \times N_{\ell-1}}$ is the weight matrix, $\mathbf{b}^\ell \in \mathbb{R}^{N_\ell}$ is the bias, and $\tanh(\cdot)$ is an element-wise activation hyperbolic tangent function.

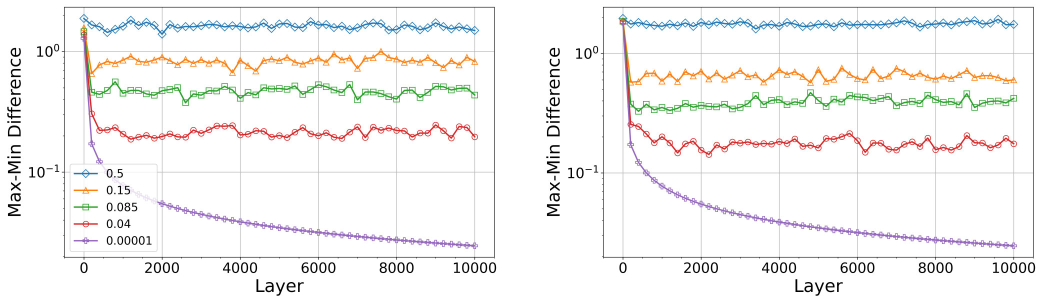
We present a simplified analysis of signal propagation in FFNNs with the tanh activation function. For notational convenience, it is assumed that all hidden layers, as well as the input and output layers, have a dimension of n, i.e., $N_\ell = n$ for all ℓ. Given an arbitrary input vector $\boldsymbol{x} = (x_1, \ldots, x_n)$, the first layer activation $\boldsymbol{x}^1 = \tanh(\mathbf{W}^1 \boldsymbol{x})$ can be expressed component-wise as:

$$x_i^1 = \tanh\left(w_{i1}^1 x_1 + \cdots + w_{in}^1 x_n\right) = \tanh\left(\left(w_{ii}^1 + \sum_{\substack{j=1 \\ j\neq i}}^{n} \frac{w_{ij}^1 x_j}{x_i}\right) x_i\right), \text{ for } i = 1, \ldots, n.$$

For the $k+1$-th layer, $i = 1, \ldots, n$, this expression can be generalized as:

$$x_i^{k+1} = \tanh\left(a_i^{k+1} x_i^k\right), \text{ where } \quad a_i^{k+1} = w_{ii}^{k+1} + \sum_{\substack{j=1 \\ j\neq i}}^{n} \frac{w_{ij}^{k+1} x_j^k}{x_i^k}. \tag{1}$$

According to Lemma 2, when $a > 1$, for an arbitrary initial value $x_0 > 0$ or $x_0 < 0$, the sequence $\{x_k\}$ defined by $x_{k+1} = \tanh(a x_k)$ converges to ξ_a or $-\xi_a$, respectively, as $k \to \infty$. This result indicates that the sequence converges to the fixed point ξ_a regardless of the initial value x_0 and ensures that the activation values do not vanish as network depth increases. Furthermore, by Lemma 2, if $a_i^k \leq 1$ for all $N \leq k \leq L$, then x_i^L approaches zero. Therefore, to ensure that (i) a_i^k remains close to 1 and (ii) $a_i^k \leq 1$ does not hold for all $N \leq k \leq L$, we design the initial weight matrix as



(a) Same dimension

(b) Varying dimensions

Figure 2: Difference between maximum and minimum activation values at each layer when propagating $3,000$ input data through a $10,000$-layer tanh FFNN, using the proposed initialization with α set to $0.04, 0.085, 0.15$, and 0.5. Network with $10,000$ hidden layers, each with 32 nodes (Left), and a network with alternating hidden layers of 64 and 32 nodes (Right).

$\mathbf{W}^\ell = \mathbf{D}^\ell + \mathbf{Z}^\ell \in \mathbb{R}^{N_\ell \times N_{\ell-1}}$, where $\mathbf{D}^\ell_{i,j} = 1$ if $i \equiv j \pmod{N_{\ell-1}}$, 0 otherwise, and $\mathbf{Z}^\ell$ is a noise matrix drawn from $\mathcal{N}(0, \sigma_z^2)$, where σ_z is set to $\alpha/\sqrt{N^{\ell-1}}$ and $\alpha = 0.085$. Then a_i^{k+1} follows the distribution:

$$a_i^{k+1} \sim \mathcal{N}\left(1,\, \sigma_z^2 + \sigma_z^2 \sum_{\substack{j=1 \\ j\neq i}}^{n} \left(\frac{x_j^k}{x_i^k}\right)^2\right). \tag{2}$$

According to Equation 2, the mean of a_i^{k+1} is 1, so choosing an appropriate σ_z satisfies condition (i). For condition (ii), if x_i^k becomes small relative to other elements in $\boldsymbol{x}^k$, the variance of a_i^{k+1} increases, as indicated by Equation 2. As a result, the probability that the absolute value of x_i^{k+1} surpasses that of x_i^k is higher. However, if σ_z is too small, the increase in the variance of a_i^{k+1} becomes limited. Therefore, choosing an appropriate σ_z is crucial.

3.3 Preventing activation saturation via appropriate σ_z tuning

In this subsection, we discuss how σ_z impacts the scale of the activation values. Equation 2 indicates that a_i^k follows a normal distribution, with variance depending on σ_z. Firstly, we experimentally investigated the impact of σ_z on the scale of the activation values. As demonstrated in Figure 2, increasing $\sigma_z = \alpha/\sqrt{N_{\ell-1}}$ causes the activation values in any layer to be distributed over a broader range. However, setting σ_z to a large value can lead to saturation, where most activations converge towards -1 and 1. If σ_z is too large, the probability that $a_i^{(k)}$ takes values far from 1 (e.g., -10, 5, etc.) increases. This, in turn, increases the value of $1+\epsilon$ mentioned in Corollary 4, potentially bounding the activation values in sufficiently deep layers by $\xi_{1+\epsilon}$. Consequently, the activation values in deeper layers become less likely to approach zero and tend to saturate toward specific values. On the other hand, if σ_z is too small, as mentioned in Subsection 3.2, the variance of a_i^k becomes restricted. This is demonstrated experimentally in Figure 2, when $\alpha = 0.00001$. For this reason, we experimentally found an optimal $\sigma_z = \alpha/\sqrt{N^{\ell-1}}$, with $\alpha = 0.085$, that is neither too large nor too small. Results from experiments solving the Burgers' equation using PINNs with varying σ_z are presented in Appendix B.1.

4 Experiments

In this section, we conduct a series of experiments to validate the proposed weight initialization method. In Subsection 4.1, we evaluate the performance of an FFNN with the tanh activation function on benchmark datasets. In Subsection 4.2, we solve the Burgers' equation and Allen-Cahn equation using Physics-Informed Neural Networks. Both experiments are conducted across various network sizes to verify whether the proposed method consistently performs well, independent of net-

Table 1: Validation accuracy and loss are presented for FFNNs with varying numbers of nodes $(2, 8, 32, 128)$, each with 20 hidden layers using the tanh activation function. All models were trained for 20 epochs, and the highest average accuracy and lowest average loss, computed from 10 runs, are presented. When comparing different initialization methods under the same experimental settings, the better-performing method is highlighted in bold. Underlined values indicate the highest accuracy when only the number of nodes is varied.

	2		8		32		128	
MNIST	Accuracy	Loss	Accuracy	Loss	Accuracy	Loss	Accuracy	Loss
Xavier	49.78	1.632	68	0.958	91.67	0.277	<u>95.45</u>	0.154
Proposed	**62.82**	**1.185**	**77.95**	**0.706**	**92.51**	**0.255**	<u>**96.12**</u>	**0.134**
FMNIST	Accuracy	Loss	Accuracy	Loss	Accuracy	Loss	Accuracy	Loss
Xavier	42.89	1.559	68.55	0.890	81.03	0.533	<u>86.2</u>	0.389
Proposed	**51.65**	**1.324**	**71.31**	**0.777**	**83.06**	**0.475**	<u>**87.12**</u>	**0.359**
CIFAR10	Accuracy	Loss	Accuracy	Loss	Accuracy	Loss	Accuracy	Loss
Xavier	32.82	1.921	43.51	1.608	<u>48.62</u>	1.473	47.58	1.510
Proposed	**38.16**	**1.780**	**47.04**	**1.505**	<u>**48.80**</u>	**1.463**	**48.51**	**1.471**

work depth and width. The experiments were conducted in TensorFlow without skip connections, normalization layers, and learning rate decay in any of the experiments.

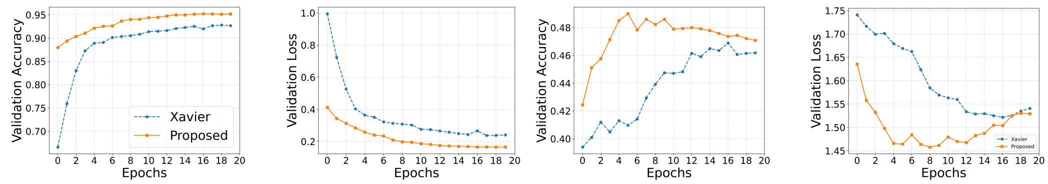
4.1 Width Independence in Classification Task

To evaluate the effectiveness of the proposed weight initialization method, we conduct experiments on the MNIST, Fashion MNIST, and CIFAR-10 (Krizhevsky & Hinton, 2009) datasets, utilizing the Adam optimizer. All experiments are conducted with a batch size of 64 and a learning rate of 0.0001. Fifteen percent of the total dataset is allocated for validation.

We apply the proposed weight initialization method to evaluate its effectiveness in training tanh FFNNs, emphasizing its robustness to variations in network width. Four tanh FFNNs are created, each with 20 hidden layers, and with 2, 8, 32, and 128 nodes per hidden layer, respectively. In Table 1, for both the MNIST and Fashion MNIST datasets, the network with 128 nodes achieves the highest accuracy and lowest loss when our proposed method is employed. However, for the CIFAR-10 dataset, the network with 32 nodes yields the highest accuracy and lowest loss when employing the proposed method. In summary, our proposed method demonstrates robustness regardless of the number of nodes in tanh FFNNs. We provide more detailed experimental results in Appendix B.2.

Table 2: Validation accuracy and loss are presented for FFNNs with varying numbers of layers $(10, 50, 100)$, each with 64 number of nodes using the tanh activation function. All models were trained for 40 epochs, and the highest average accuracy and lowest average loss, computed from 10 runs, are presented.

	10		50		100	
MNIST	Accuracy	Loss	Accuracy	Loss	Accuracy	Loss
Xavier	96.55	0.112	<u>96.57</u>	0.123	94.08	0.194
Proposed	<u>**97.04**</u>	**0.102**	**96.72**	**0.109**	**96.06**	**0.132**
FMNIST	Accuracy	Loss	Accuracy	Loss	Accuracy	Loss
Xavier	<u>88.73</u>	0.319	87.72	0.344	83.41	0.463
Proposed	<u>**89.42**</u>	**0.305**	**88.51**	**0.324**	**86.01**	**0.382**
CIFAR10	Accuracy	Loss	Accuracy	Loss	Accuracy	Loss
Xavier	<u>48.39</u>	1.468	47.87	1.474	46.71	1.503
Proposed	**48.41**	**1.458**	**48.71**	**1.461**	<u>**48.96**</u>	**1.437**



(a) MNIST Accuracy (b) MNIST Loss (c) CIFAR10 Accuracy (d) CIFAR10 Loss

Figure 3: Validation accuracy and loss for a tanh FFNN with 60 hidden layers, where the number of nodes alternates between 32 and 16 across layers, repeated 30 times. The model was trained for 20 epochs on the MNIST and CIFAR-10 datasets.

4.2 Depth Independence in Classification Task

It is well known that the expressivity of neural networks generally increases exponentially with depth, enabling strong generalization performance (Poole et al., 2016; Raghu et al., 2017). Therefore, we employ the proposed weight initialization method to investigate its effectiveness in training deep FFNNs with the tanh activation function, emphasizing its robustness to variations in network depth. We create three tanh FFNNs, each with 64 nodes in all hidden layers, but with 10, 50, and 100 hidden layers, respectively. In Table 2, for both the MNIST and Fashion MNIST datasets, the network with 10 hidden layers achieves the highest accuracy and lowest loss when our proposed method is employed. Both initialization methods perform best in networks with the fewest layers, with performance degrading as the depth increases. However, for the CIFAR-10 dataset, we observe that the performance of the proposed method improves as the number of layers increases.

Furthermore, we conduct experiments with varying hidden layer dimensions, as shown in Figure 3. The network consists of 60 hidden layers, where the number of nodes alternates between 32 and 16 in each layer. We demonstrate superior performance in terms of both loss and accuracy across all epochs on the MNIST and CIFAR-10 datasets.

4.3 Network size independence in PINN

Xavier initialization is the primary method used for training PINNs (Jin et al., 2021; Son et al., 2023; Yao et al., 2023; Gnanasambandam et al., 2023). In this section, we experimentally demonstrate that the method's training performance is highly dependent on randomness and network size. Additionally, empirical results are provided demonstrating that the proposed method is more robust to variations in network size.

All experiments on Physics-Informed Neural Networks (PINNs) use full-batch training with a learning rate of 0.001. In this section, we solve the Allen-Cahn and Burgers' equations using a tanh FFNN-based PINN with 20,000 collocation points. For the Allen-Cahn equation, the diffusion coefficient is set to $d = 0.01$. The initial condition is defined as $u(x, 0) = x^2 \cos(\pi x)$ for $x \in [-1, 1]$, with boundary conditions $u(-1, t) = -1$ and $u(1, t) = -1$, applied over the time interval $t \in [0, 1]$. Similarly, for the Burgers' equation, a viscosity coefficient of $\nu = 0.01$ is employed. The initial condition is given by $u(x, 0) = -\sin(\pi x)$ for $x \in [-1, 1]$, with boundary conditions $u(-1, t) = 0$ and $u(1, t) = 0$ imposed for $t \in [0, 1]$.

The Allen-Cahn equation is expressed as:

$$\frac{\partial u}{\partial t} - d\frac{\partial^2 u}{\partial x^2} = -\frac{u^3 + u}{d}$$

where $u(x, t)$ represents the solution, d is the diffusion coefficient, and the nonlinear term $u^3 - u$ models the phase separation dynamics.

The Burgers' equation is given by:

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = \nu\frac{\partial^2 u}{\partial x^2}$$

where $u(x, t)$ is the velocity field, and ν is the viscosity coefficient.

Eight tanh FFNNs are created, each with 16 nodes in all hidden layers, but with 5, 10, 20, 30, 40, 50, 60, and 80 hidden layers, respectively. As shown in Table 3, for the Allen-Cahn equation, Xavier initialization achieves the lowest loss at a network depth of 20. However, as the depth increases, the loss gradually rises. In contrast, the proposed method achieves the lowest loss at a depth of 50 and maintains a loss of 0.00057 even at a depth of 80 layers. For the Burgers' equation, the proposed method achieves the lowest loss at a depth of 60, while at the same depth, Xavier initialization results in a loss that is an order of magnitude higher (approximately 10^2 difference).

Next, we double the number of nodes to observe the impact of node size on the loss. Eight new tanh FFNNs are created, each with 32 nodes in all hidden layers, and with 5, 10, 20, 30, 40, 50, 60, and 80 hidden layers, respectively. As shown in Table 3, for the Allen-Cahn equation, Xavier initialization achieves the lowest loss at a network depth of 30. However, as the depth increases, the model becomes untrainable, with a loss of 0.694 at a depth of 80. In contrast, the proposed method achieves the lowest loss at a depth of 40 and maintains a loss of 0.00059 even at a depth of 80 layers. For the Burgers' equation, both methods show similar loss values up to a depth of 30. Beyond a depth of 40, however, the loss steadily increases with the Xavier method, while the proposed method records the lowest loss at a depth of 50.

Table 3: A PINN loss is presented for FFNNs with varying numbers of layers $(5, 10, 20, 30, 40, 50, 60, 80)$ using the tanh activation function. The top table shows results with 16 nodes per layer, and the bottom table shows results with 32 nodes per layer. All models were trained for 300 iterations using Adam and 300 iterations using L-BFGS. The median PINN loss from the final iteration for the Burgers and Allen–Cahn equations, computed over 5 runs, is presented.

Allen-Cahn (16 Nodes)	5	10	20	30	40	50	60	80
Xavier	9.58e-04	8.16e-04	7.61e-04	1.06e-03	1.1e-03	1.24e-03	3.55e-03	1.81e-03
Proposed	**9.21e-04**	**7.29e-04**	**5.76e-04**	**5.29e-04**	**5.37e-04**	**4.03e-04**	**4.73e-04**	**5.77e-04**
Burgers (16 Nodes)	5	10	20	30	40	50	60	80
Xavier	6.97e-03	1.11e-02	7.9e-03	9.71e-03	2.45e-02	2.65e-02	6.5e-02	5.71e-02
Proposed	**6.19e-03**	**5.08e-03**	**5.28e-03**	**9.31e-04**	**3.56e-03**	**8.27e-04**	**3.43e-04**	**2.05e-03**

Allen-Cahn (32 Nodes)	5	10	20	30	40	50	60	80
Xavier	3.13e-01	5.03e-02	3.64e-03	2.37e-03	4.03e-03	5.27e-03	1.73e-02	6.94e-01
Proposed	**1.04e-03**	**6.92e-04**	**5.34e-04**	**4.26e-04**	**3.31e-04**	**3.52e-04**	**3.85e-04**	**5.96e-04**
Burgers (32 Nodes)	5	10	20	30	40	50	60	80
Xavier	1.12e-02	**3.53e-03**	2.72e-03	1.81e-03	7.60e-03	8.56e-03	9.86e-03	1.66e-01
Proposed	**4.14e-03**	4.11e-03	**1.58e-03**	**1.29e-03**	**7.96e-04**	**5.85e-04**	**9.80e-04**	**1.47e-03**

5 Conclusion

In this paper, we have introduced a novel weight initialization method for tanh FFNNs, grounded in the theoretical analysis of fixed points of the $\tanh(ax)$ function. Through our fixed-point analysis, we established conditions under which the vanishing or exploding of activation values can be prevented, even as the depth of the network increases.

Our proposed method exhibits strong robustness to variations in network size, as demonstrated across a variety of FFNN configurations and benchmark datasets, including MNIST, Fashion MNIST, and CIFAR-10. In contrast to Xavier initialization, which struggles to maintain stable performance as network depth increases, the proposed method consistently achieves superior results by preserving activation values. Furthermore, we explored the impact of the initialization hyperparameter σ_z on the distribution of activation values. We demonstrated both theoretically and experimentally that the choice of σ_z plays a significant role in maintaining the proper range of activations, balancing between vanishing and saturation. In the context of PINNs, the proposed initialization method shows improved performance in solving PDEs such as the Burgers' equation and the Allen-Cahn equation. By maintaining a stable loss function and achieving faster convergence compared to

Xavier initialization, our method demonstrates its practical utility in training networks for physical systems.

A key advantage of the proposed method lies in its robustness to network depth and width, significantly reducing the need for extensive hyperparameter tuning. By maintaining stable performance across varying network configurations, our approach helps to minimize the time and effort spent on searching for optimal network architectures, allowing researchers to focus on model design and other aspects of the training process. This makes the proposed method particularly valuable in large-scale and resource-constrained applications where efficient training is critical.

REFERENCES

Thomas Bachlechner, Bodhisattwa Prasad Majumder, Henry Mao, Gary Cottrell, and Julian McAuley. Rezero is all you need: fast convergence at large depth. In Cassio de Campos and Marloes H. Maathuis (eds.), *Proceedings of the Thirty-Seventh Conference on Uncertainty in Artificial Intelligence*, volume 161 of *Proceedings of Machine Learning Research*, pp. 1352–1361. PMLR, 27–30 Jul 2021.

Hassan Bararnia and Mehdi Esmaeilpour. On the application of physics informed neural networks (pinn) to solve boundary layer thermal-fluid problems. *International Communications in Heat and Mass Transfer*, 132:105890, 2022. ISSN 0735-1933. doi: https://doi.org/10.1016/j.icheatmasstransfer.2022.105890.

Yoshua Bengio, Paolo Frasconi, and Patrice Simard. The problem of learning long-term dependencies in recurrent networks. In *IEEE international conference on neural networks*, pp. 1183–1188. IEEE, 1993.

Salvatore Cuomo, Vincenzo Schiano Di Cola, Fabio Giampaolo, Gianluigi Rozza, Maziar Raissi, and Francesco Piccialli. Scientific machine learning through physics–informed neural networks: Where we are and what's next. *Journal of Scientific Computing*, 92(3):88, 2022a. doi: https://doi.org/10.1007/s10915-022-01939-z.

Salvatore Cuomo, Vincenzo Schiano Di Cola, Fabio Giampaolo, Gianluigi Rozza, Maziar Raissi, and Francesco Piccialli. Scientific machine learning through physics–informed neural networks: Where we are and what's next. *Journal of Scientific Computing*, 92(3):88, 2022b.

G. Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, 2(4):303–314, 1989. URL `https://doi.org/10.1007/BF02551274`.

Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 249–256. JMLR Workshop and Conference Proceedings, 2010.

Raghav Gnanasambandam, Bo Shen, Jihoon Chung, Xubo Yue, and Zhenyu Kong. Self-scalable tanh (stan): Multi-scale solutions for physics-informed neural networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023.

G. Gripenberg. Approximation by neural networks with a bounded number of nodes at each level. *Journal of Approximation Theory*, 122(2):260–266, 2003. ISSN 0021-9045. doi: https://doi.org/10.1016/S0021-9045(03)00078-9.

Namig J. Guliyev and Vugar E. Ismailov. Approximation capability of two hidden layer feedforward neural networks with fixed weights. *Neurocomputing*, 316:262–269, 2018a. ISSN 0925-2312. doi: https://doi.org/10.1016/j.neucom.2018.07.075.

Namig J. Guliyev and Vugar E. Ismailov. On the approximation by single hidden layer feedforward neural networks with fixed weights. *Neural Networks*, 98:296–304, 2018b. ISSN 0893-6080. doi: https://doi.org/10.1016/j.neunet.2017.12.007.

John M. Hanna, José V. Aguado, Sebastien Comas-Cardona, Ramzi Askri, and Domenico Borzacchiello. Residual-based adaptivity for two-phase flow simulation in porous media using physics-informed neural networks. *Computer Methods in Applied Mechanics and Engineering*, 396:115100, 2022. ISSN 0045-7825. doi: https://doi.org/10.1016/j.cma.2022.115100.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pp. 1026–1034, 2015.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Kurt Hornik. Approximation capabilities of multilayer feedforward networks. *Neural Networks*, 4 (2):251–257, 1991. ISSN 0893-6080. doi: https://doi.org/10.1016/0893-6080(91)90009-T.

Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366, 1989. ISSN 0893-6080. doi: https://doi.org/10.1016/0893-6080(89)90020-8.

Vahid Reza Hosseini, Abbasali Abouei Mehrizi, Afsin Gungor, and Hamid Hassanzadeh Afrouzi. Application of a physics-informed neural network to solve the steady-state bratu equation arising from solid biofuel combustion theory. *Fuel*, 332:125908, 2023. ISSN 0016-2361. doi: https://doi.org/10.1016/j.fuel.2022.125908.

Ameya D. Jagtap, Ehsan Kharazmi, and George Em Karniadakis. Conservative physics-informed neural networks on discrete domains for conservation laws: Applications to forward and inverse problems. *Computer Methods in Applied Mechanics and Engineering*, 365:113028, 2020. ISSN 0045-7825. doi: https://doi.org/10.1016/j.cma.2020.113028. URL https://www.sciencedirect.com/science/article/pii/S0045782520302127.

Ameya D Jagtap, Zhiping Mao, Nikolaus Adams, and George Em Karniadakis. Physics-informed neural networks for inverse problems in supersonic flows. *Journal of Computational Physics*, 466:111402, 2022.

Ameya Dilip Jagtap and George E. Karniadakis. Extended physics-informed neural networks (xpinns): A generalized space-time domain decomposition based deep learning framework for nonlinear partial differential equations. *Communications in Computational Physics*, 2020. URL https://api.semanticscholar.org/CorpusID:229083388.

Xiaowei Jin, Shengze Cai, Hui Li, and George Em Karniadakis. Nsfnets (navier-stokes flow nets): Physics-informed neural networks for the incompressible navier-stokes equations. *Journal of Computational Physics*, 426:109951, 2021.

George Em Karniadakis, Ioannis G Kevrekidis, Lu Lu, Paris Perdikaris, Sifan Wang, and Liu Yang. Physics-informed machine learning. *Nature Reviews Physics*, 2021.

Diederik P Kingma. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. 2009.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

Hyunwoo Lee, Yunho Kim, Seung Yeop Yang, and Hayoung Choi. Improved weight initialization for deep and narrow feedforward neural network. *Neural Networks*, 176:106362, 2024.

Dong C Liu and Jorge Nocedal. On the limited memory bfgs method for large scale optimization. *Mathematical programming*, 45(1):503–528, 1989.

Xu Liu, Xiaoya Zhang, Wei Peng, Weien Zhou, and Wen Yao. A novel meta-learning initialization method for physics-informed neural networks. *Neural Computing and Applications*, 34(17): 14511–14534, 2022.

Lu Lu, Yeonjong Shin, Yanhui Su, and George Em Karniadakis. Dying relu and initialization: Theory and numerical examples. *arXiv preprint arXiv:1903.06733*, 2019.

Lu Lu, Pengzhan Jin, Guofei Pang, Zhongqiang Zhang, and George Em Karniadakis. Learning nonlinear operators via deeponet based on the universal approximation theorem of operators. *Nature machine intelligence*, 2021.

Vitaly Maiorov and Allan Pinkus. Lower bounds for approximation by mlp neural networks. *Neurocomputing*, 25(1):81–91, 1999. ISSN 0925-2312. doi: https://doi.org/10.1016/S0925-2312(98)00111-8.

Zhiping Mao, Ameya D. Jagtap, and George Em Karniadakis. Physics-informed neural networks for high-speed flows. *Computer Methods in Applied Mechanics and Engineering*, 360:112789, 2020. ISSN 0045-7825. doi: https://doi.org/10.1016/j.cma.2019.112789.

Dmytro Mishkin and Jiri Matas. All you need is a good init. In *ICLR*, 2016.

Sejun Park, Chulhee Yun, Jaeho Lee, and Jinwoo Shin. Minimum width for universal approximation. *CoRR*, abs/2006.08859, 2020. URL `https://arxiv.org/abs/2006.08859`.

Pai Peng, Jiangong Pan, Hui Xu, and Xinlong Feng. Rpinns: Rectified-physics informed neural networks for solving stationary partial differential equations. *Computers & Fluids*, 245:105583, 2022. ISSN 0045-7930. doi: https://doi.org/10.1016/j.compfluid.2022.105583.

Ben Poole, Subhaneil Lahiri, Maithra Raghu, Jascha Sohl-Dickstein, and Surya Ganguli. Exponential expressivity in deep neural networks through transient chaos. *Advances in neural information processing systems*, 29, 2016.

Maithra Raghu, Ben Poole, Jon Kleinberg, Surya Ganguli, and Jascha Sohl-Dickstein. On the expressive power of deep neural networks. In *ICML*, pp. 2847–2854, 2017.

Maziar Raissi, Paris Perdikaris, and George E Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational physics*, 378:686–707, 2019.

Pratik Rathore, Weimu Lei, Zachary Frangella, Lu Lu, and Madeleine Udell. Challenges in training PINNs: A loss landscape perspective. In *ICML*, 2024.

David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.

Andrew M Saxe, James McClelland, and Surya Ganguli. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. In *ICLR*, 2014.

Zuowei Shen, Haizhao Yang, and Shijun Zhang. Optimal approximation rate of relu networks in terms of width and depth. *Journal de Mathématiques Pures et Appliquées*, 157:101–135, 2022. ISSN 0021-7824. doi: https://doi.org/10.1016/j.matpur.2021.07.009.

Khemraj Shukla, Patricio Clark Di Leoni, James Blackshire, Daniel Sparkman, and George Em Karniadakis. Physics-informed neural network for ultrasound nondestructive quantification of surface breaking cracks. *Journal of Nondestructive Evaluation*, 39(3):61, 2020. doi: 10.1007/s10921-020-00705-1. URL `https://doi.org/10.1007/s10921-020-00705-1`.

Khemraj Shukla, Ameya D. Jagtap, and George Em Karniadakis. Parallel physics-informed neural networks via domain decomposition. *Journal of Computational Physics*, 447:110683, 2021. ISSN 0021-9991. doi: https://doi.org/10.1016/j.jcp.2021.110683. URL `https://www.sciencedirect.com/science/article/pii/S0021999121005787`.

Hwijae Son, Sung Woong Cho, and Hyung Ju Hwang. Enhanced physics-informed neural networks with augmented lagrangian relaxation method (al-pinns). *Neurocomputing*, 548:126424, 2023.

Yanjie Song, He Wang, He Yang, Maria Luisa Taccari, and Xiaohui Chen. Loss-attentional physics-informed neural networks. *Journal of Computational Physics*, 501:112781, 2024.

Homayoon Tarbiyati and Behzad Nemati Saray. Weight initialization algorithm for physics-informed neural networks using finite differences. *Engineering with Computers*, pp. 1–17, 2023.

Zhiyong Wu, Huan Wang, Chang He, Bing J. Zhang, Tao Xu, and Qinglin Chen. The application of physics-informed machine learning in multiphysics modeling in chemical engineering. *Industrial & Engineering Chemistry Research*, 62, 2023. ISSN 18178-18204.

Zixue Xiang, Wei Peng, Xu Liu, and Wen Yao. Self-adaptive loss balanced physics-informed neural networks. *Neurocomputing*, 496:11–34, 2022. ISSN 0925-2312. doi: https://doi.org/10.1016/j.neucom.2022.05.015.

Liu Yang, Xuhui Meng, and George Em Karniadakis. B-pinns: Bayesian physics-informed neural networks for forward and inverse pde problems with noisy data. *Journal of Computational Physics*, 425:109913, 2021. ISSN 0021-9991. doi: https://doi.org/10.1016/j.jcp.2020.109913.

Jiachen Yao, Chang Su, Zhongkai Hao, Songming Liu, Hang Su, and Jun Zhu. Multiadam: Parameter-wise scale-invariant optimizer for multiscale training of physics-informed neural networks. In *International Conference on Machine Learning*, pp. 39702–39721. PMLR, 2023.

Dmitry Yarotsky. Error bounds for approximations with deep relu networks. *Neural Networks*, 94:103–114, 2017. ISSN 0893-6080. doi: https://doi.org/10.1016/j.neunet.2017.07.002.

Minglang Yin, Xiaoning Zheng, Jay D. Humphrey, and George Em Karniadakis. Non-invasive inference of thrombus material properties with physics-informed neural networks. *Computer Methods in Applied Mechanics and Engineering*, 375:113603, 2021. ISSN 0045-7825. doi: https://doi.org/10.1016/j.cma.2020.113603. URL https://www.sciencedirect.com/science/article/pii/S004578252030788X.

Jeremy Yu, Lu Lu, Xuhui Meng, and George Em Karniadakis. Gradient-enhanced physics-informed neural networks for forward and inverse pde problems. *Computer Methods in Applied Mechanics and Engineering*, 393:114823, 2022. ISSN 0045-7825. doi: https://doi.org/10.1016/j.cma.2022.114823. URL https://www.sciencedirect.com/science/article/pii/S0045782522001438.

Lei Yuan, Yi-Qing Ni, Xiang-Yun Deng, and Shuo Hao. A-pinn: Auxiliary physics informed neural networks for forward and inverse problems of nonlinear integro-differential equations. *Journal of Computational Physics*, 462:111260, 2022. ISSN 0021-9991. doi: https://doi.org/10.1016/j.jcp.2022.111260.

Jiawei Zhao, Florian Tobias Schaefer, and Anima Anandkumar. Zero initialization: Initializing neural networks with only zeros and ones. *Transactions on Machine Learning Research*, 2022. ISSN 2835-8856.

Jing'ang Zhu, Yiheng Xue, and Zishun Liu. A transfer learning enhanced physics-informed neural network for parameter identification in soft materials. *Applied Mathematics and Mechanics*, 45 (10):1685–1704, 2024. doi: 10.1007/s10483-024-3178-9. URL https://doi.org/10.1007/s10483-024-3178-9.

A PROOFS OF THE THEORETICAL RESULTS

A.1 PROOF OF LEMMA 1

Proof. We define $g(x) = \tanh(ax) - x$. Since $g(x)$ is continuous, and $g(-M) > 0$, $g(M) < 0$ for a large real number $M \in \mathbb{R}^+$, the Intermediate Value Theorem guarantees the existence of a point x such that $g(x) = 0$.

First, consider the case $0 < a \leq 1$. Since $0 < a \leq 1$, the derivative $g'(x) = a \cdot \operatorname{sech}^2(ax) - 1$ satisfies $-1 \leq g'(x) \leq a - 1 < 0$ for all x. Hence, $g(x)$ is strictly decreasing and therefore $g(x)$ has the unique root. At $x = 0$, $\phi(0) = \tanh(a \cdot 0) = 0$. Hence, $x = 0$ is the unique fixed point.

Let us consider the case $a > 1$. For $0 < x \ll 1$, $\tanh(ax) - x \approx (a-1)x$. Since $a > 1$, $\tanh(ax) - x > 0$. On the other hand, since $|\tanh(ax)| < 1$ for all x,

$$\lim_{x\to\infty}[-1 - x] \leq \lim_{x\to\infty}[\tanh(ax) - x] \leq \lim_{x\to\infty}[1 - x].$$

By the squeeze theorem, $\lim_{x\to\infty}[\tanh(ax) - x] = -\infty$. By the intermediate value theorem, therefore, there exists at least one $x > 0$ such that $\tanh(ax) = x$. To establish the uniqueness of the positive fixed point, we investigate the derivative $g'(x) = a \operatorname{sech}^2(ax) - 1$. We find the critical points to be $x = \pm\frac{1}{a}\sec^{-1}(\frac{1}{\sqrt{a}})$. It is straightforward to see that $g'(x) > 0$ in $\left(-\frac{1}{a}\sec^{-1}(\frac{1}{\sqrt{a}}), \frac{1}{a}\sec^{-1}(\frac{1}{\sqrt{a}})\right)$ and $g'(x) < 0$ in $\mathbb{R}\setminus\left(-\frac{1}{a}\sec^{-1}(\frac{1}{\sqrt{a}}), \frac{1}{a}\sec^{-1}(\frac{1}{\sqrt{a}})\right)$. i.e. $g(x) = 0$ has exactly two fixed points. Because $g(x)$ is an odd function, if x^* is a solution, then $-x^*$ is also a solution. Thus, for $a > 1$, there exists a unique positive fixed point if $x > 0$ and a unique negative fixed point if $x < 0$. $\square$

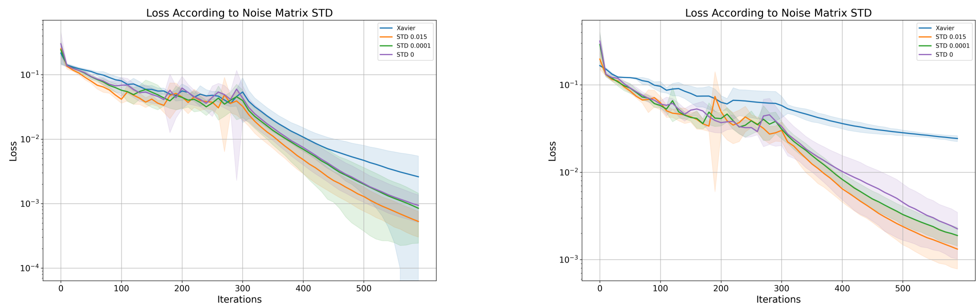
A.2 PROOF OF LEMMA 2

Proof. (1) Since $(\tanh(ax))' = a \operatorname{sech}^2(ax) < 1$ for all $x > 0$, it holds that $x_{n+1} = \phi_a(x_n) < x_n$ for all $n \in \mathbb{N}$. Thus the sequence $\{x_n\}_{n=1}^{\infty}$ is decreasing. Since $x_n > 0$ for all $n \in \mathbb{N}$, by the monotone convergence theorem, it converges to the fixed point $x^* = 0$.

(2) Let $x_0 < \xi_a$. Since $\phi'(x)$ decreasing for $x \geq 0$, with $\phi'(0) > 1$ and ξ_a is the unique fixed point for $x > 0$, it holds that $x_n < x_{n+1} < \xi_a$ for all $n \in \mathbb{N}$. Thus, by the monotone convergence theorem, the sequence converges to the fixed point ξ_a. The proof is similar when $x_0 > \xi_a$. $\square$

B ADDITIONAL EXPERIMENTAL RESULTS

B.1 PREVENTING ACTIVATION SATURATION VIA APPROPRIATE σ_z TUNING



(a) Same dimension

(b) Varying dimensions

Figure 4: Here, 'STD' refers to σ_z. (a) shows the PINN loss for the Burgers' equation, using an FFNN with 30 layers and 32 nodes in each hidden layer. (b) shows the PINN loss for an FFNN with 30 layers, where the hidden layers alternate between 64 and 32 nodes, repeated 15 times. Each experiment was repeated 10 times with different random seeds.

B.2 WIDTH INDEPENDENCE IN CLASSIFICATION TASKS

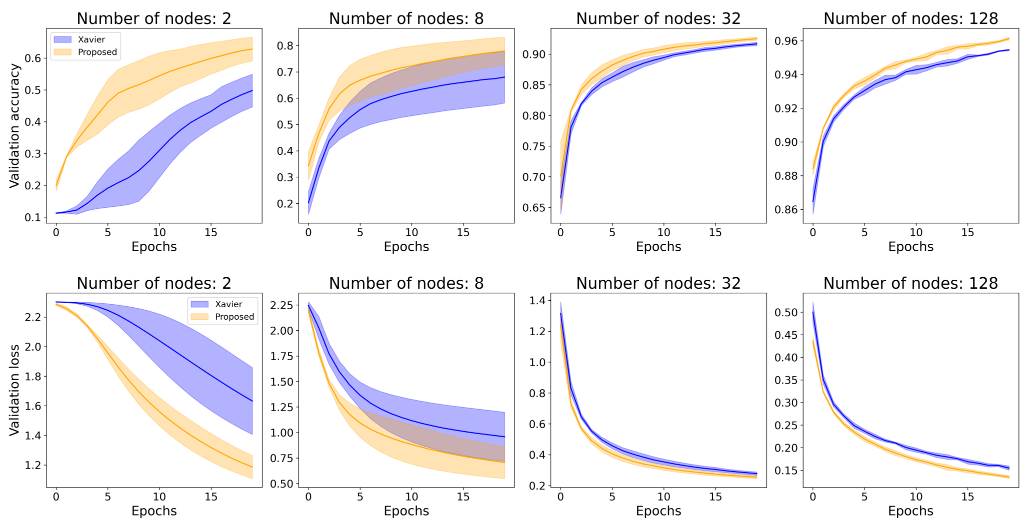


Figure 5: Validation accuracy and loss are presented for tanh FFNNs with varying numbers of nodes $(2, 8, 32, 128)$, each with 20 hidden layers. All models were trained for 20 epochs on the MNIST dataset, with 10 different random seeds.

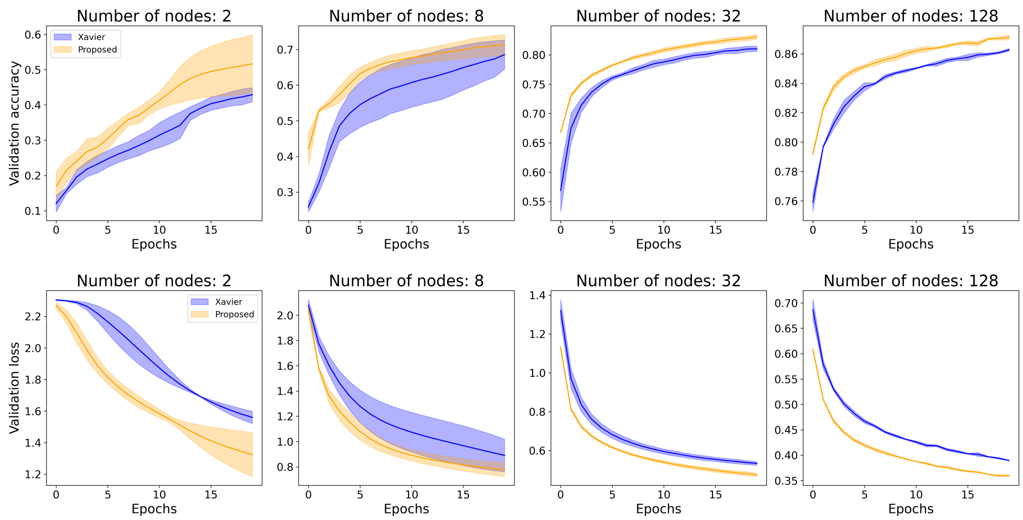


Figure 6: Validation accuracy and loss are presented for tanh FFNNs with varying numbers of nodes $(2, 8, 32, 128)$, each with 20 hidden layers. All models were trained for 20 epochs on the Fashion MNIST dataset, with 10 different random seeds.

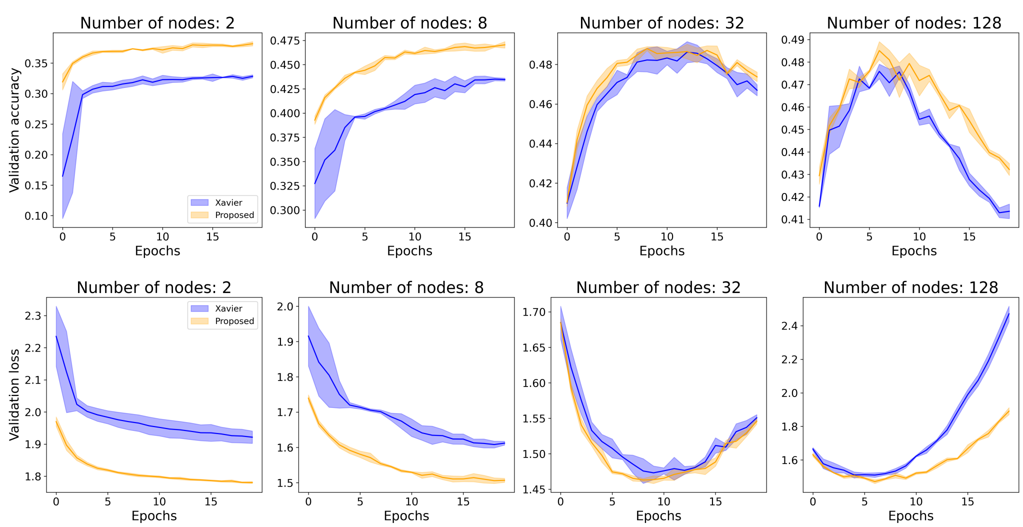


Figure 7: Validation accuracy and loss are presented for tanh FFNNs with varying numbers of nodes $(2, 8, 32, 128)$, each with 20 hidden layers. All models were trained for 20 epochs on the CIFAR-10 dataset, with 10 different random seeds.