Beyond Gaussian Initializations: Signal Preserving Weight Initialization for Odd-Sigmoid Activations

Hyunwoo Lee

Center for AI and Natural Sciences
Korea Institute for Advanced Study

January 7, 2026

Contents

1. Introduction

Preliminary

- Let K pairs of training samples $\{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^{K}$, where $\mathbf{x}_i \in \mathbb{R}^{N_x}$ is the training input and $\mathbf{y}_i \in \mathbb{R}^{N_y}$ is its corresponding output.
- Here, N_x and N_y are the number of nodes in the input layer and output layer, respectively.

Feedforward Neural Network (FFNN)

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

where $\mathbf{x}^{\ell-1} \in \mathbb{R}^{N_{\ell-1}}$ is the input feature of ℓ-th layer, $\mathbf{W}^{\ell} \in \mathbb{R}^{N_\ell \times N_{\ell-1}}$ is the weight matrix, $\mathbf{b}^{\ell} \in \mathbb{R}^{N_\ell}$ is the bias vector for each $\ell = 1, \ldots, L$, and $f(\cdot)$ is an element-wise activation function.

Weight Initialization

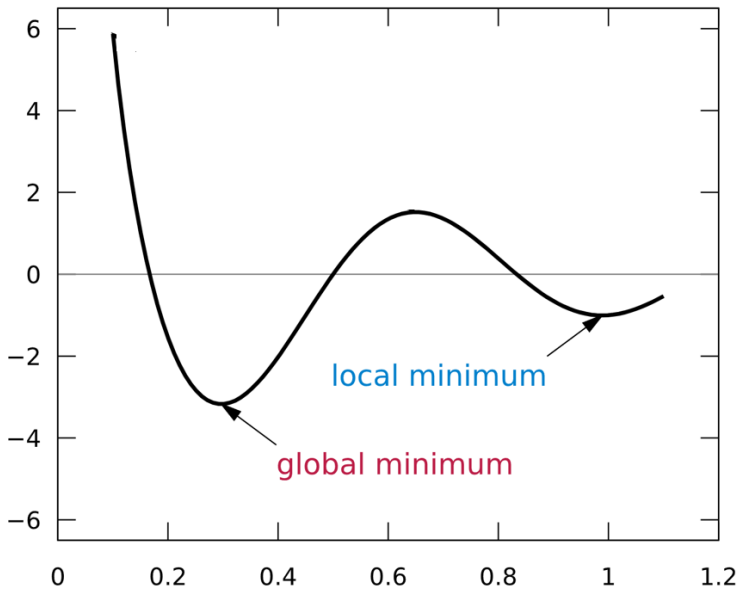
The weight initialization aims to set the initial values of $\{\mathbf{W}^{\ell}\}_{\ell=1}^{L}$ such that training is efficient and convergence is achieved.

Preliminary

Question

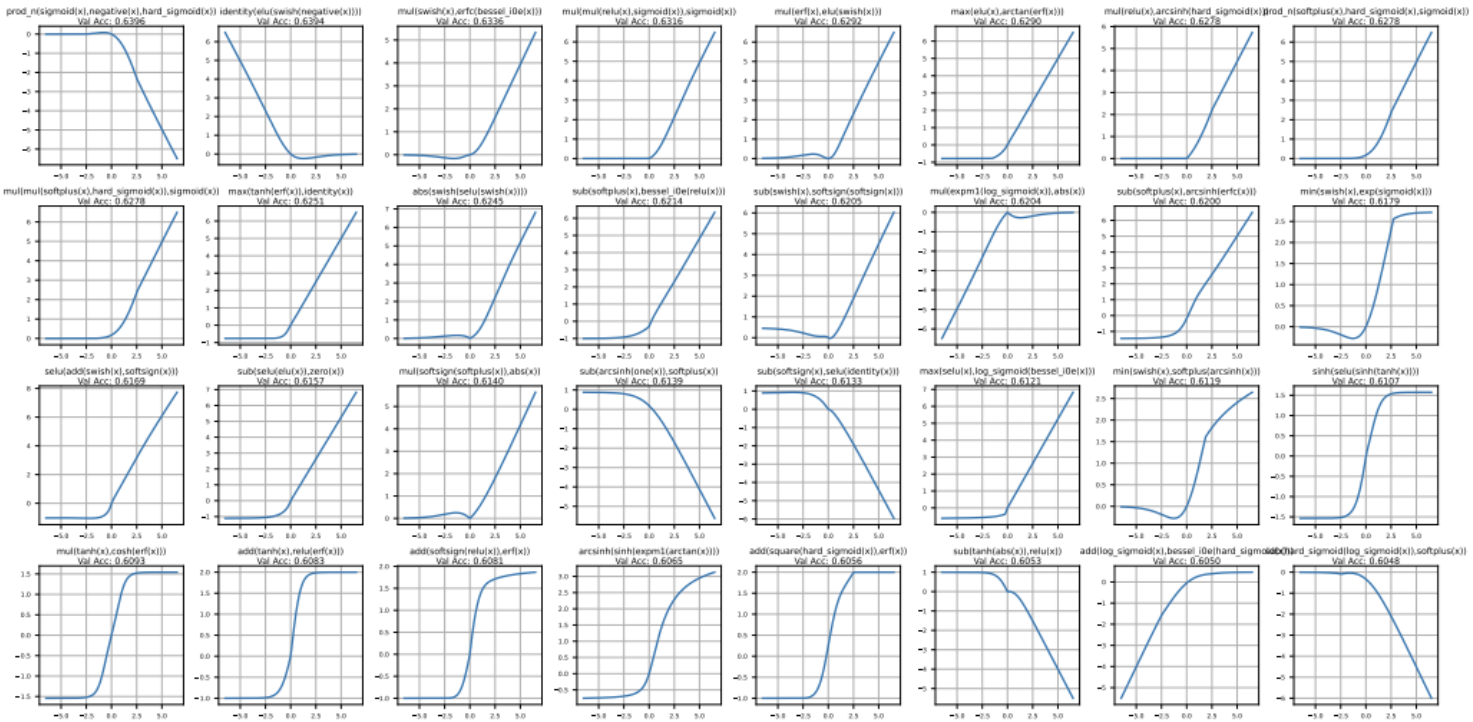
Why is weight initialization important for training neural networks [1, 2]?

- Speeds Up Convergence
- Improves Model Performance
- Dataset Efficiency
- Enables Training Across Various Architecture Sizes



Motivation

- Neural networks need diverse nonlinearities [21].
- Prior work evaluated 2,913 activation functions [22].



Motivation

$-x \cdot \sigma(x) \cdot \text{HardSigmoid}(x)$	**0.7776**
$\text{Swish}(x)/\text{SELU}(1)$	0.7771
$\text{Swish}(x) \cdot \text{erfc}(\text{bessel\_i0e}(x))$	0.7755
$\sigma(\text{Softsign}(x)) \cdot \text{ELU}(x)$	0.7734
$\text{SELU}(\sinh(e^{\arctan(x)} - 1))$	0.7719
$\text{HardSigmoid}(\text{HardSigmoid}(x)) \cdot \text{ELU}(x)$	0.7718
$\text{ELU}(\text{Swish}(-x))$	0.7679
$\text{Swish}(-2x)$	0.7664
$x \cdot \text{erfc}(\text{ELU}(x))$	0.7635
$\text{ReLU}(x)$	0.7660

Could we explore a wider range of activation functions?

Activation functions and **weight initialization** are interdependent.

Overview of Weight Initialization Methods

Weight Initialization for ReLU Neural Networks

- Orthogonal initialization (Saxe et al., 2014) [13]
- **He initialization** (K. He et al., 2015) [12]
- Gaussian submatrix initialization (R. Burkholz et al., 2019) [11]
- Randomized asymmetric initialization (L. Lu et al., 2020) [10]
- Zero initialization (J. Zhao et al., 2022) [14]
- Improved initialization (H, Lee, et al., 2024) [15]

Weight Initialization for Sigmoidal Neural Networks

- **Xavier initialization** (X. Glorot & Y. Bengio, 2010) [9]
- **EOC initialization** (S, Hayou, et al., 2019) [25]
- Robust initialization (H, Lee, et al., 2024) [20]

Motivation

Activation-aware initialization

- Activation functions and initialization are interdependent.
- Under standard initializations, trainable activations are limited.
- Accordingly, we propose an activation-aware initialization.

Activation families

Activation functions are broadly categorized into two families: **sigmoidal** and **ReLU like**.

Motivation

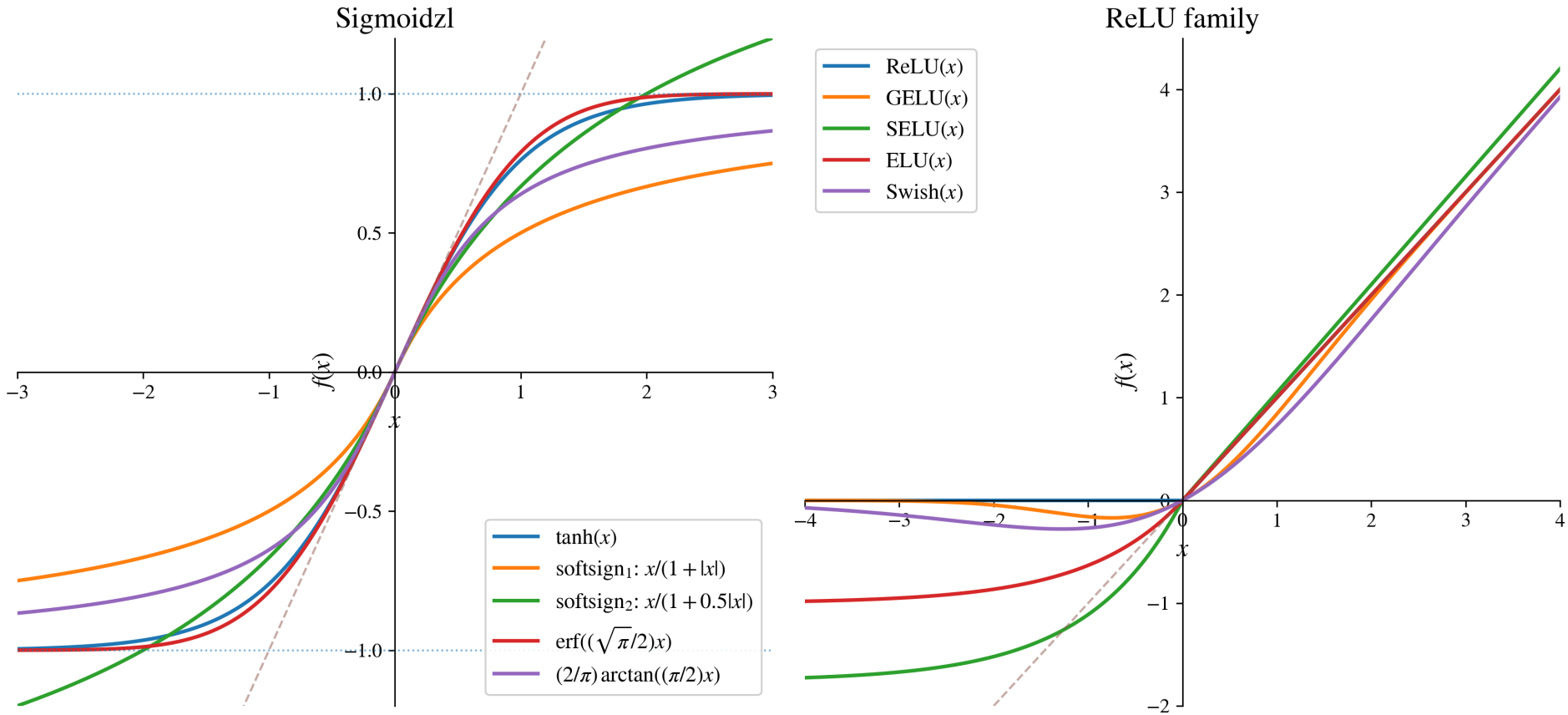


Figure 1: Sigmoidal, ReLU like

Odd-Sigmoid Function

Definition. Odd-Sigmoid Function

A function $f : \mathbb{R} \to \mathbb{R}$ is an **odd-sigmoid function** if it satisfies:

(i) **Regularity:** $f \in C^1(\mathbb{R})$.

(ii) **Odd symmetry:** $f(-x) = -f(x)$ for all $x \in \mathbb{R}$.

(iii) **Boundedness:** $\sup_{x \in \mathbb{R}} |f(x)| < \infty$.

(iv) **Strict monotonicity:** $f'(x) > 0$ for all $x \in \mathbb{R}$.

(v) **Slope decay:** f' is strictly decreasing on $[0, \infty)$.

Denote the class of all odd-sigmoid functions by $\mathcal{F}$.

Odd-Sigmoid Function

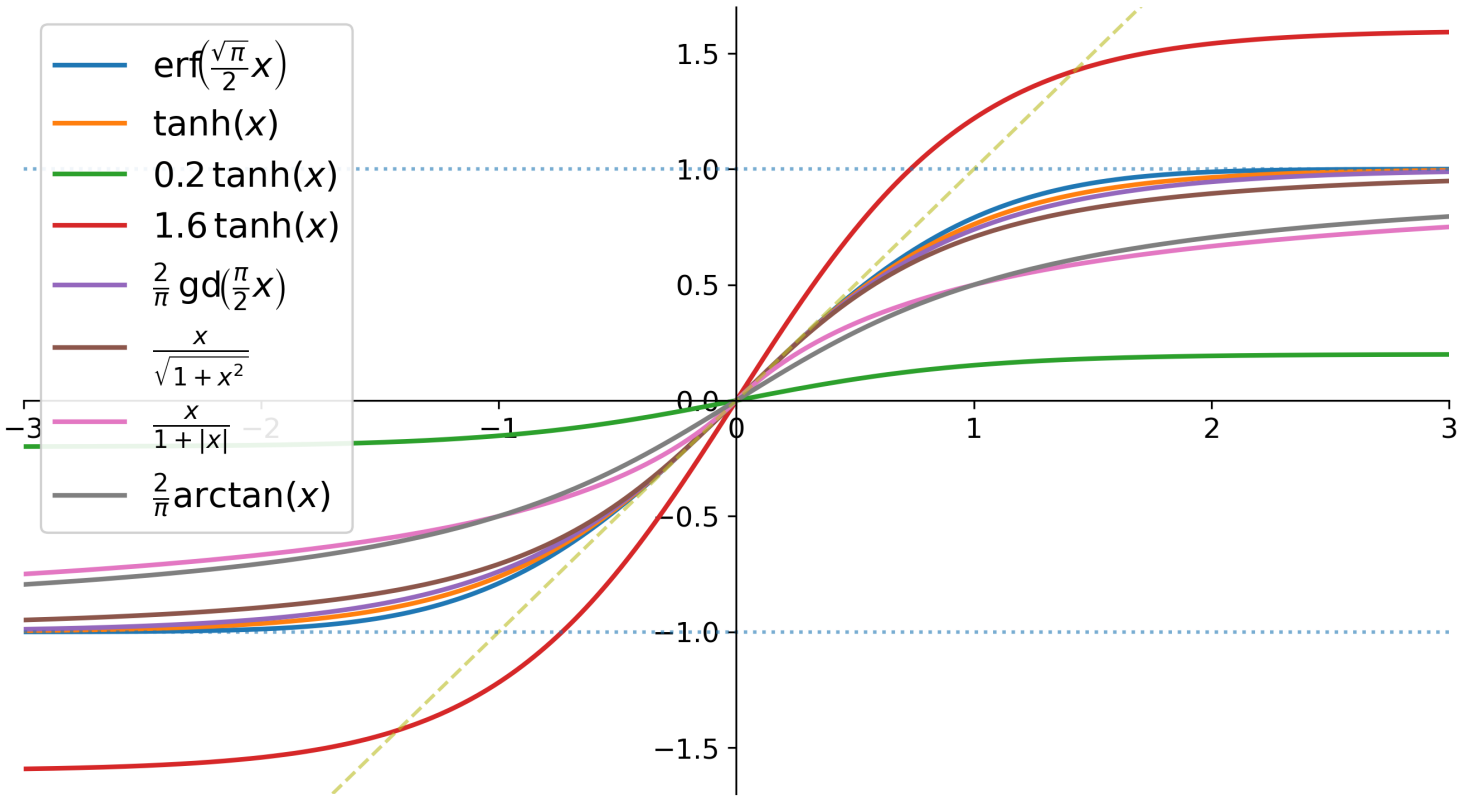


Figure 2: Examples of odd-sigmoid function.

Prior Method

Edge of Chaos (EOC) [24]

Mean-field theory for i.i.d. Gaussian initialization $w_{ij}^{\ell} \sim \mathcal{N}(0, \sigma_w^2/N_\ell)$ yields an average gradient amplification factor

$$\chi_{\ell+1} \approx \sigma_w^2 \, \mathbb{E}\left[f'\left(\mathbf{h}^{\ell+1}\right)^2 \right].$$

If $\chi_{\ell+1} < 1$, gradients vanish (ordered phase); if $\chi_{\ell+1} > 1$, they explode (chaotic phase). The *edge of chaos* is the critical curve $\chi_{\ell+1} \approx 1$, enabling deep signal/gradient propagation.

EOC Initialization [25]

Choose Gaussian i.i.d. initialization parameters (σ_w^2, σ_b^2) on the EOC curve to improve information propagation and often accelerate training.

Overview: Beyond Gaussian Initializations

Goal

1. **Gaussian initializations:** Xavier, LeCun, He, and EOC, etc.
2. **Goal:** overcome Gaussian initialization limits.

Limitations of Gaussian Initializations

1. Training can fail in **deep and narrow** networks.
2. Performance is **sensitive to the choice of** σ.
3. Training can be difficult across **diverse activation functions**.

Our Approach

1. We propose an initialization for a defined function class, based on a **pitchfork bifurcation**, without assuming wide networks.
2. We connect σ_z to the **sign-flip probability**.
3. We enable training even for **very large-scale activations**, such as $10^{10}\tanh(x)$.

2. Proposed Method

The Derivation of the Proposed Method

Question

How to effectively propagate signals to deeper layers in a network?

Simplified analysis of signal propagation in FFNNs

Given an arbitrary input vector $\boldsymbol{x} = (x_1, \ldots, x_n)$, the first layer activation $\boldsymbol{x}^1 = f(\mathbf{W}^1\boldsymbol{x})$ can be expressed component-wise as:

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

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

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

The Derivation of the Proposed Method

Theorem 1.

Suppose $f \in \mathcal{F}$ with $\omega := 1/f'(0)$, and for a fixed $a > 0$ define

$$x_0 > 0, \qquad x_{n+1} = f(ax_n), \qquad n = 0, 1, 2, \ldots.$$

Then the sequence $\{x_n\}$ converges for every $x_0 > 0$. Furthermore,

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

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

The Derivation of the Proposed Method

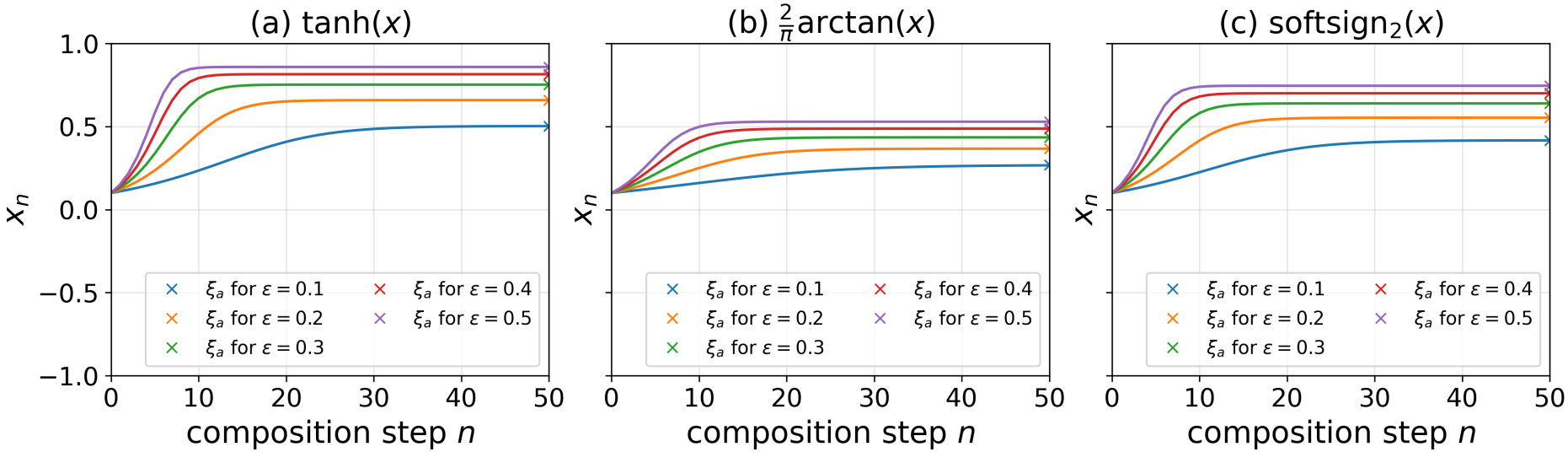


Figure 3: Convergence of the iteration $x_{n+1} = f(ax_n)$ for odd–sigmoid f with $a = \omega + \epsilon$ ($\epsilon \in \{0.1, 0.2, 0.3, 0.4, 0.5\}$); curves show x_n up to $n = 50$ from $x_0 = 0.1$, and '$\times$' marks the positive fixed point ξ_a solving $f(a\xi_a) = \xi_a$.

The Derivation of the Proposed Method

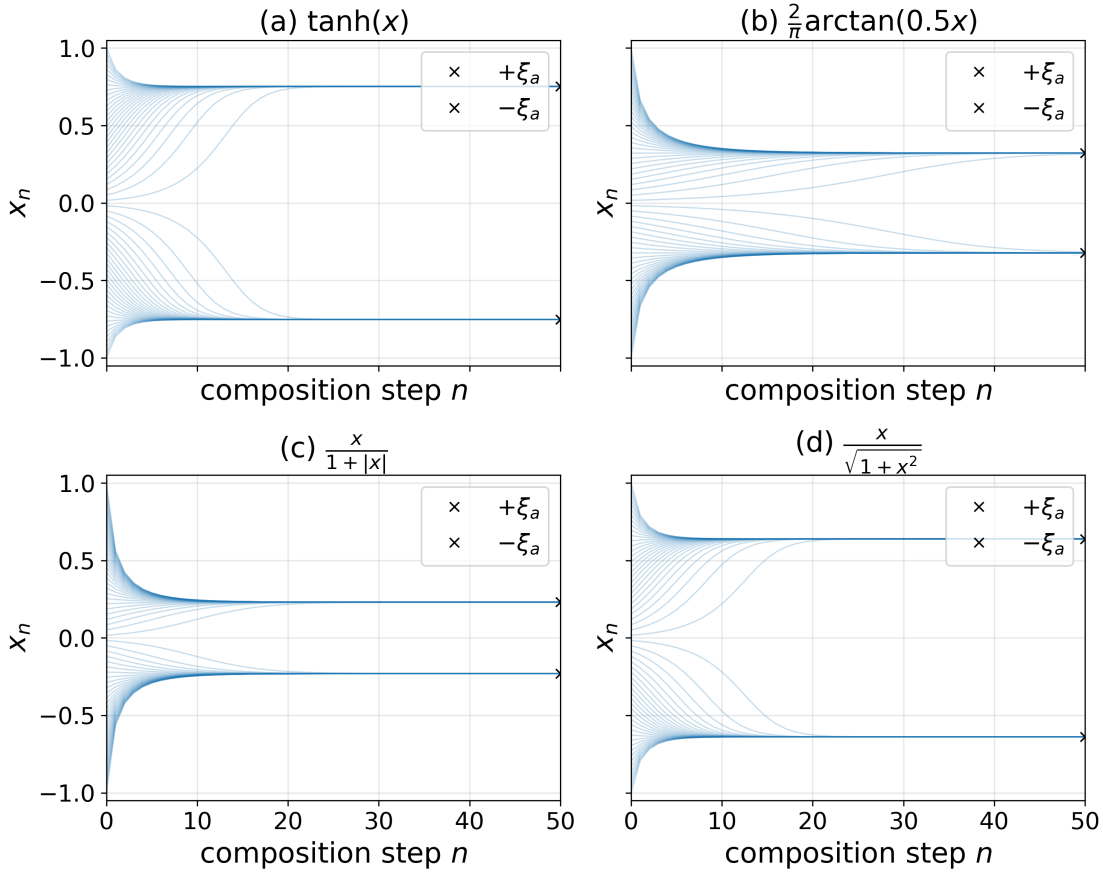


Figure 4: For each activation f, we iterate $x_{n+1} = f(ax_n)$ with $a = \omega + 0.3$ and $\omega = 1/f'(0)$ from 60 initial values $x_0 \in [-1, 1]$ up to $n = 50$; trajectories are shown (thin lines), and the fixed points $\pm\xi_a$ satisfying $f(a\xi_a) = \xi_a$ are marked at $n = 50$ with '$\times$'.

The Derivation of the Proposed Method

Theorem 2.

Let $f \in \mathcal{F}$ be an odd–sigmoid activation with $\omega := 1/f'(0)$, and fix any $\varepsilon > 0$. Consider the feedforward network and the proposed initialization, except that the diagonal element is set to $a_0 := \omega + \varepsilon$, and let $a_i^{\ell+1}$ be the effective gain. Fix a tolerance $\gamma \in (0, 1)$ and a finite depth $L \in \mathbb{N}$. Then there exist a threshold depth $L_0 \leq L$ and a noise threshold $\sigma_0 > 0$ such that, for all $0 < \sigma_z \leq \sigma_0$,

$$\mathbb{P}\Big((1-\gamma)\,\sigma_z^2 \leq \mathrm{Var}\big(a_i^{\ell+1} \mid x^\ell\big) \leq (1+\gamma)\,\sigma_z^2$$

$$\forall\, L_0 \leq \ell < L,\ 1 \leq i \leq N_\ell\Big) \geq 1-\gamma.$$

Proposed Weight Initialization

Structured + Noise initialization

Let $f \in \mathcal{F}$ and $\omega := 1/f'(0) > 0$. For each layer ℓ,

$$\mathbf{W}^\ell = \mathbf{D}^\ell + \mathbf{Z}^\ell \in \mathbb{R}^{N_\ell \times N_{\ell-1}}, \qquad (\mathbf{D}^\ell)_{ij} = \begin{cases} \omega, & i \equiv j \pmod{N_{\ell-1}}, \\ 0, & \text{otherwise}, \end{cases}$$

and

$$(\mathbf{Z}^\ell)_{ij} \sim \mathcal{N}\left(0, \frac{\sigma_z^2}{N_{\ell-1}}\right).$$

- $\mathbf{D}^\ell$ stabilizes signal propagation.
- $\mathbf{Z}^\ell$ provides flexibility for learning.

Proposed Weight Initialization

Closed-form calibration of σ_z (target depth L)

Set the target sign flip rate

$$p(L) = \begin{cases} p_{\text{real}}(= 0.4), & L \le L_{\text{th}}(= 10), \\ 2.05\, e^{-0.133L}, & L > L_{\text{th}}. \end{cases}$$

Then calibrate the noise by

$$\sigma_z := \sigma^*(p(L), L, \omega) = -\frac{\omega}{\Phi^{-1}\left(\frac{1-(1-2p(L))^{1/L}}{2}\right)},$$

where Φ is the standard normal cumulative distribution function.

Learning rate band (Adam)

$$\eta \in \left[10^{-5}\omega,\ 10^{-3}\omega\right].$$

Proposed Weight Initialization

Lemma 1.

Using the elementwise formulation and employing the proposed weight initialization, fix an arbitrary layer ℓ and index i such that $x_i^\ell \neq 0$. Then, conditionally on x^ℓ,

$$a_i^{\ell+1} \sim \mathcal{N}\left(\omega, \frac{\sigma_z^2}{N_\ell}\left(1 + \sum_{j \neq i}\left(\frac{x_j^\ell}{x_i^\ell}\right)^2\right)\right),$$

and $\mathrm{Var}(a_i^{\ell+1}) \geq \sigma_z^2 / N_\ell$.

We propose a **surrogate model** corresponding to the real model in Lemma 1.

Proposed Weight Initialization

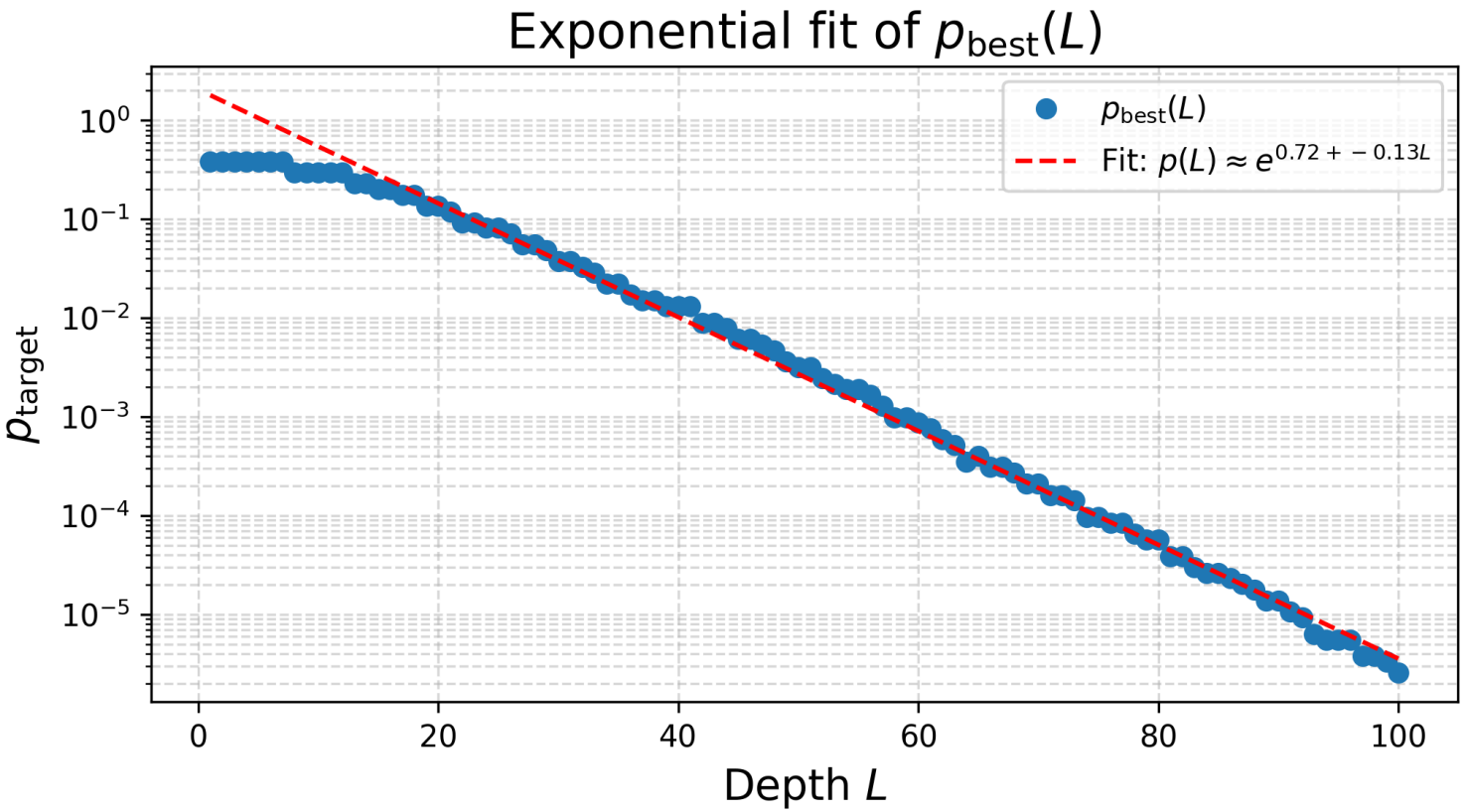
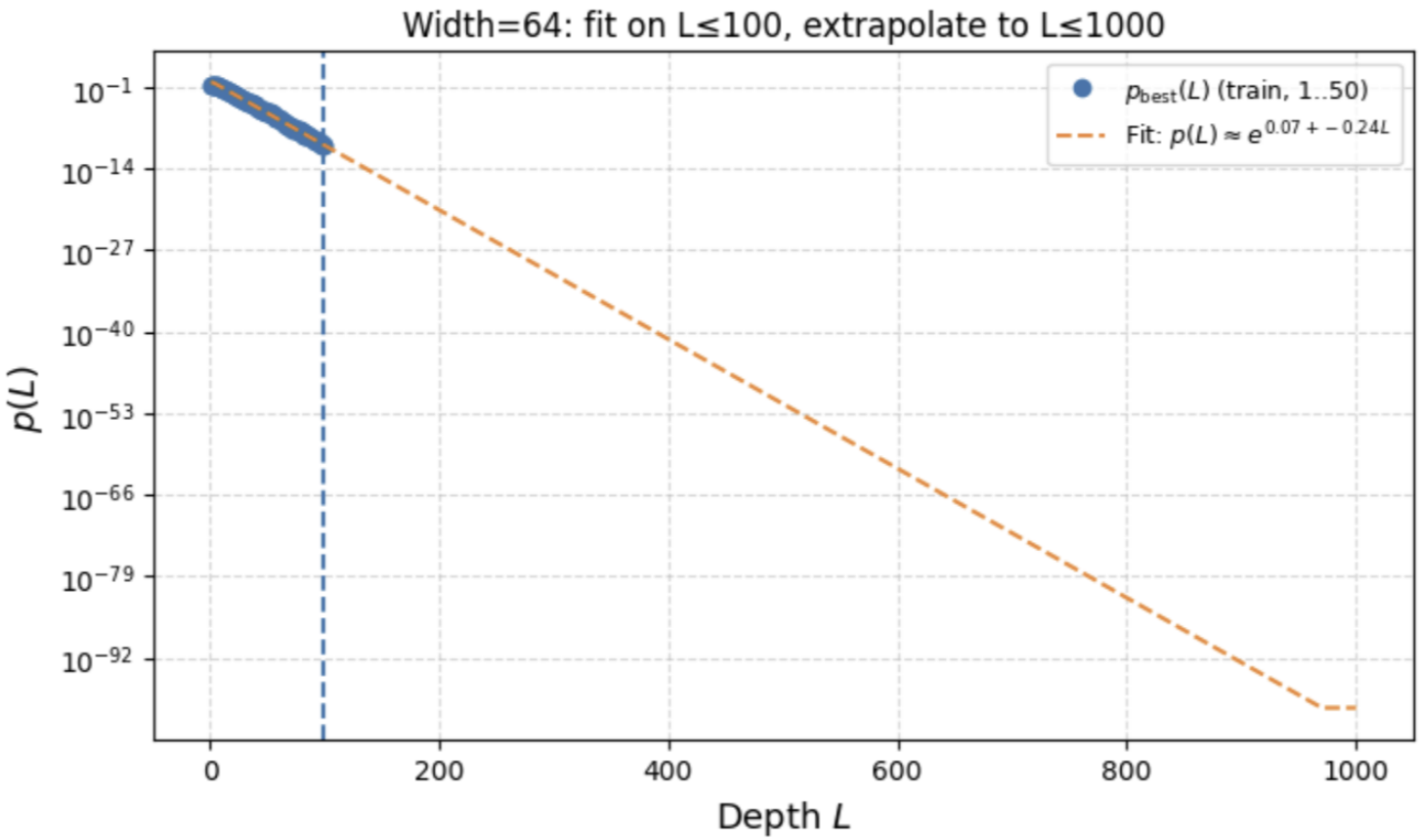
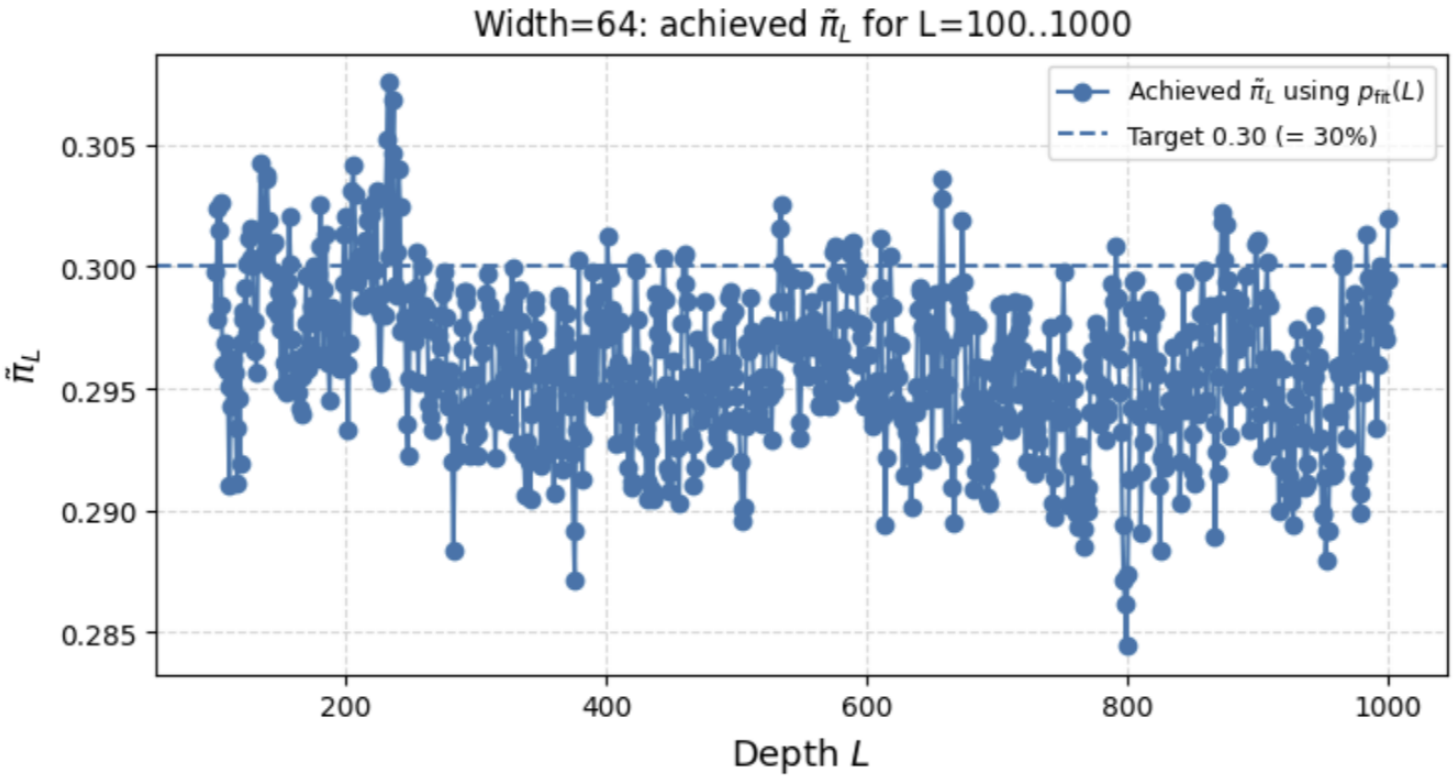


Figure 5: Best surrogate target $p_{\text{target}}(L)$ producing an FFNN-driven negative rate $\tilde{\pi}_L \approx 0.4$ as a function of depth L for a tanh network with width 64. The curve shows that $p_{\text{target}}(L)$ stays near 0.3–0.4 for shallow depths and then decays approximately exponentially with L, providing an empirical calibration rule for choosing the surrogate sign flip rate in deep networks.

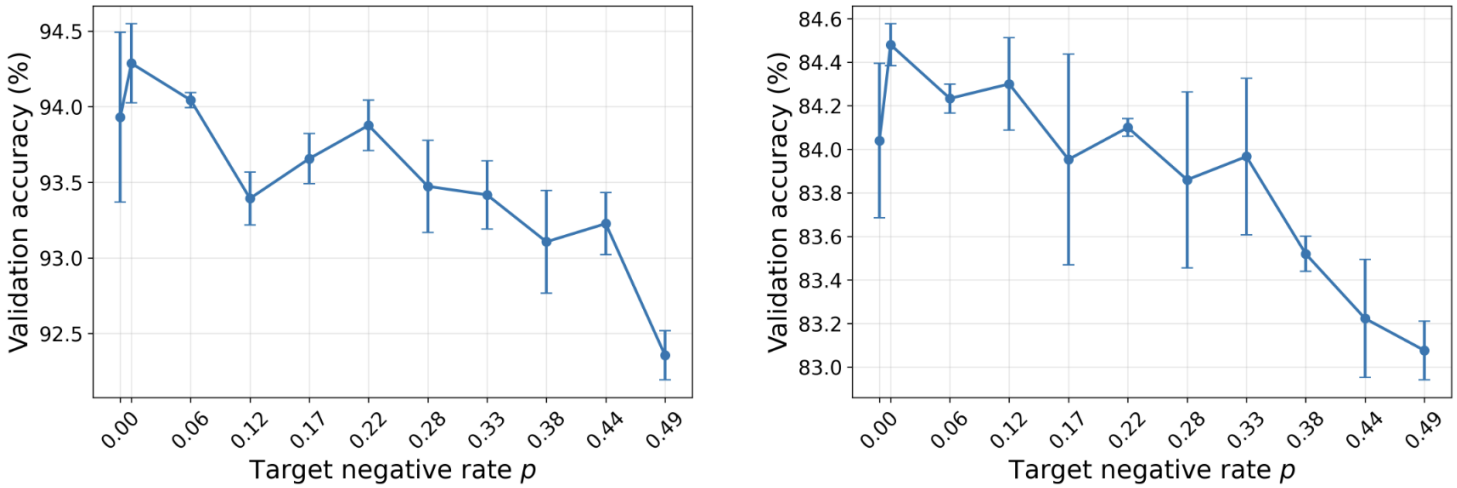
Proposed Weight Initialization



Proposed Weight Initialization



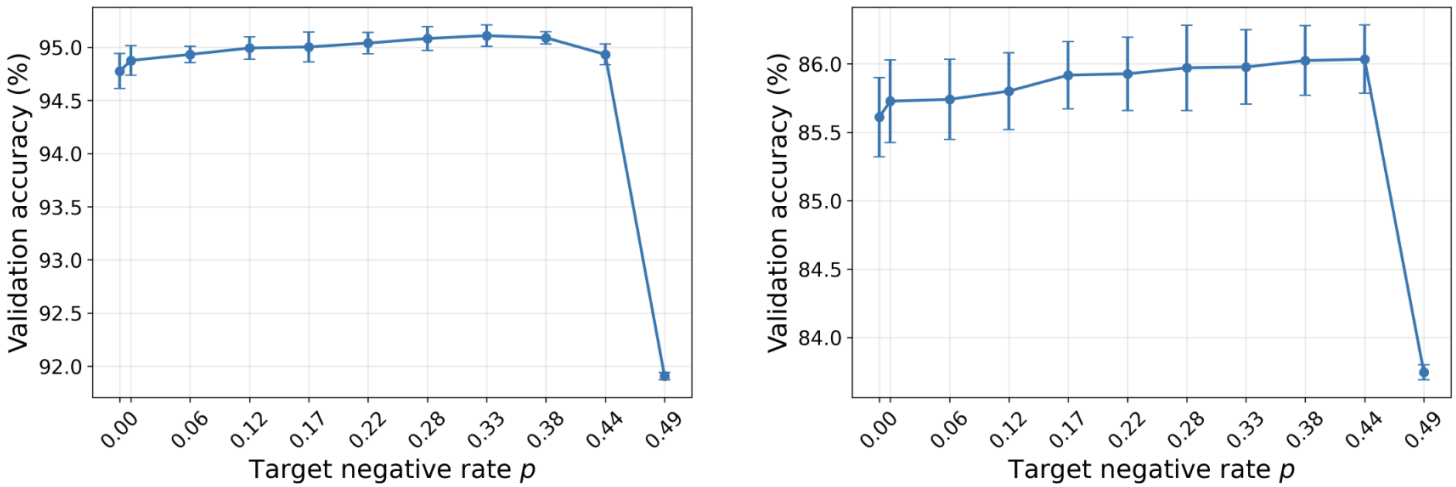
Proposed Weight Initialization



(a) MNIST (b) MNIST

Figure 6: Validation accuracy as a function of the target negative rate p for a 50-layer fully connected network (width 64) with activation $f(x) = \tanh(x)$ under the proposed initialization. Each point shows the mean $\pm$ standard deviation over 5 runs, where each run is trained for 600 iterations, with $\sigma^*(p, L, \omega)$ computed from the scalar surrogate calibration.

Proposed Weight Initialization



(a) MNIST

(b) FMNIST

Figure 7: Validation accuracy as a function of the target negative rate p for a 3-layer fully connected network (width 512) with activation $f(x) = \tanh(x)$ under the proposed initialization. Each point shows the mean $\pm$ standard deviation over 5 runs, where each run is trained for 600 iterations, with $\sigma^*(p, L, \omega)$ computed from the scalar surrogate calibration.

Comparing Gaussian and Proposed Initializations

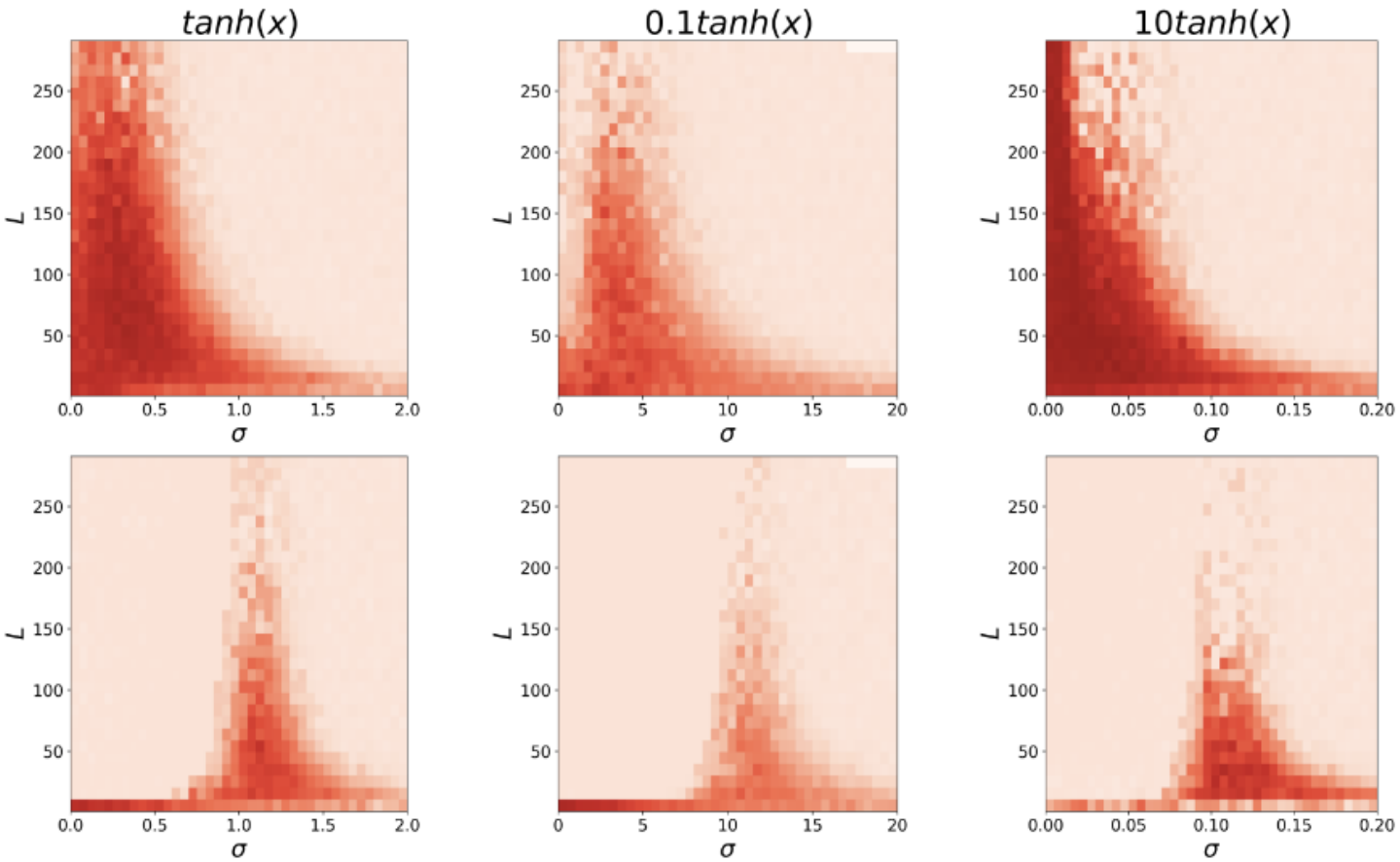


Figure 8: Heatmaps of validation accuracy on MNIST as a function of depth L and noise scale σ for three activations $\tanh(x)$, $0.1\tanh(x)$, and $10\tanh(x)$. Each model is a FFNN with width 128 trained with Adam. Each cell shows the mean validation accuracy over 3 runs. **(Top row)** Proposed. **(Bottom row)** EOC.

Comparing Gaussian and Proposed Initializations

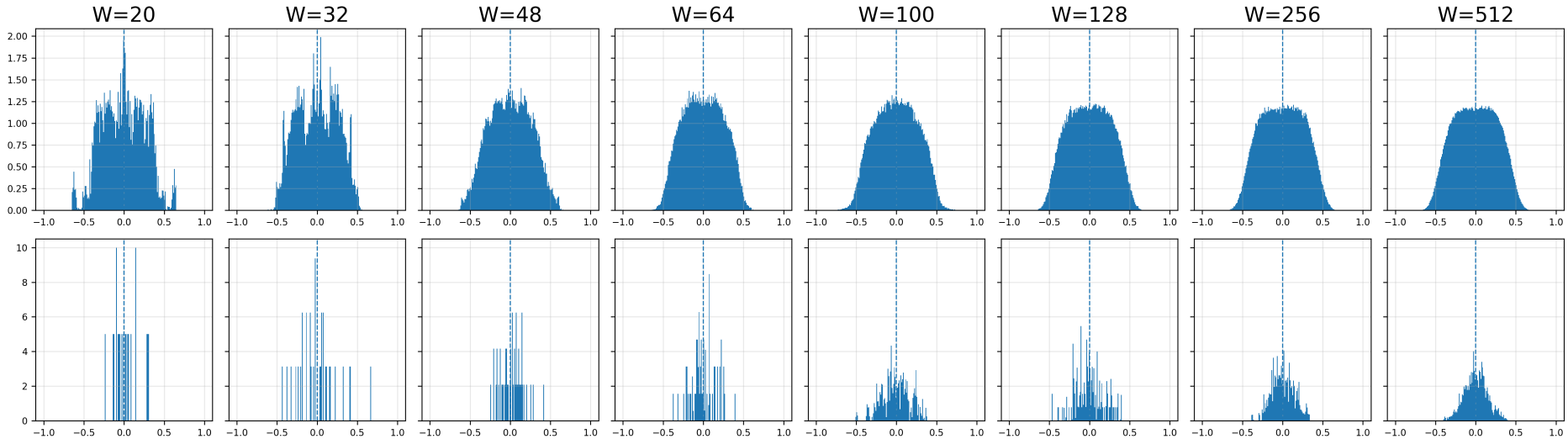
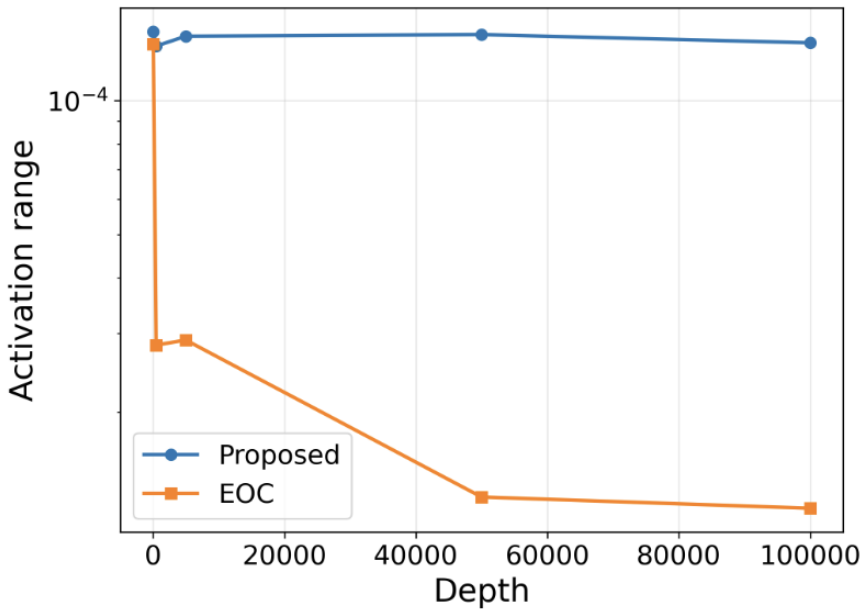
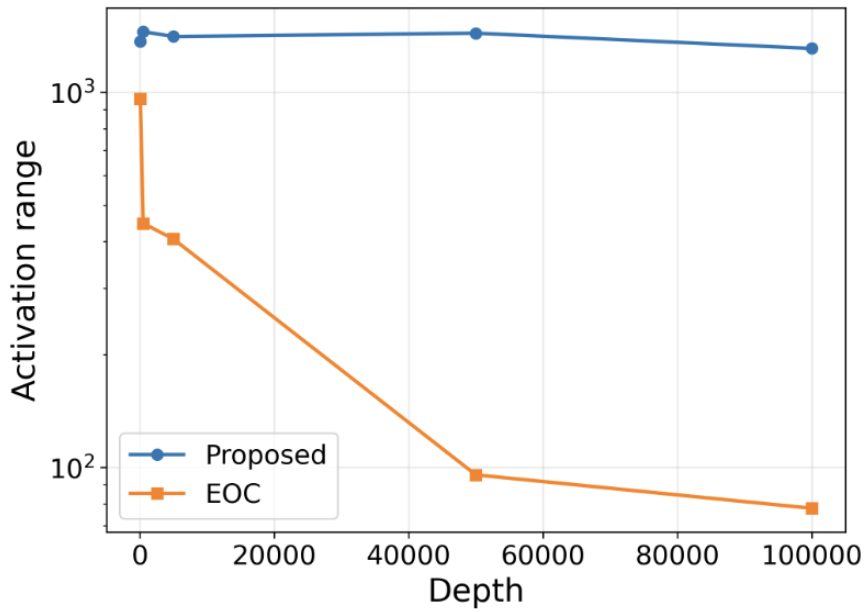


Figure 9: Last layer activation histograms for tanh networks with depth $L = 1000$ and varying width under the proposed initialization **(top row)** and the EOC initialization **(bottom row)**. Each column corresponds to a different hidden width.

Comparing Gaussian and Proposed Initializations



(a) $\alpha = 0.0001$



(b) $\alpha = 1000$

Figure 10: Activation range as a function of depth $L \in \{50, 500, 5000, 50000, 10^5\}$ for fully connected width 64 networks with activation $\alpha \tanh(x)$ under the Proposed and EOC initializations. Panel (a) uses $\alpha = 10^{-4}$ and panel (b) uses $\alpha = 10^3$.

Comparing Gaussian and Proposed Initializations

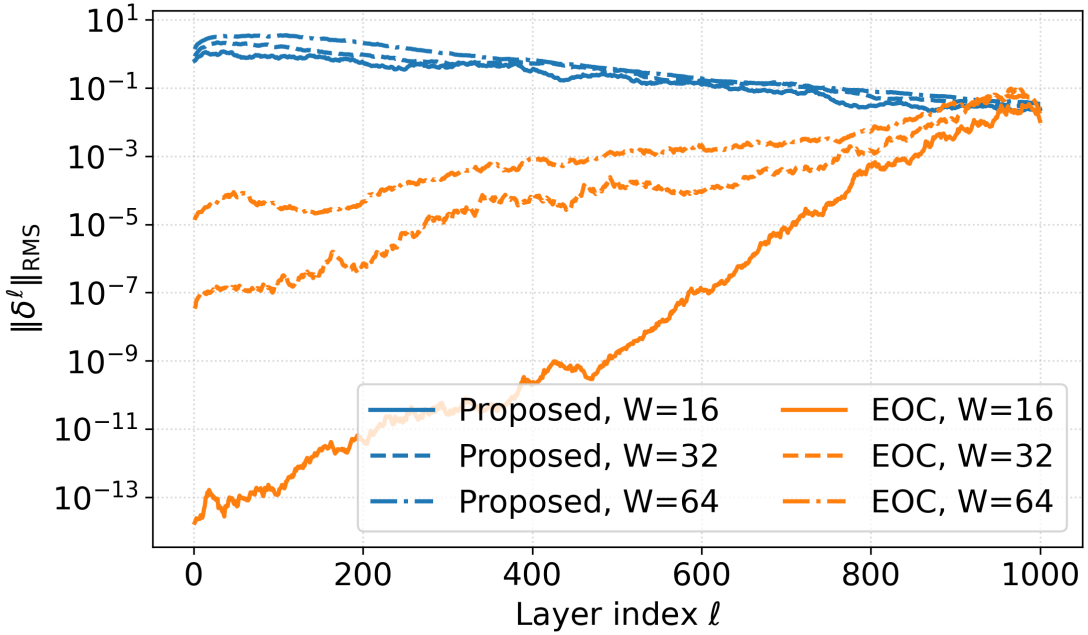


Figure 11: Layerwise gradient norms at initialization for a tanh FFNN of depth $L = 1000$. We set $\boldsymbol{\delta}^{\ell} := \partial L / \partial \mathbf{h}^{\ell}$.

3. Computational Results

Experiments with Activations $\omega \approx 1$ (Dataset Efficiency)

Table 1: Mean of the best validation accuracies within 50 epochs over 10 independent runs for a 50 layer FFNN (512 units) on MNIST and Fashion MNIST, trained on 100 or 500 sample subsets.

Dataset	Method	$\tanh(x)$		$\mathrm{erf}(x)$		$\arctan(x)$		$\mathrm{gd}(x)$		$\mathrm{softsign}_2(x)$		$\mathrm{softsign}_1(x) + \mathrm{softsign}_2(x)$	
		100	500	100	500	100	500	100	500	100	500	100	500
MNIST	Xavier	64.00	79.85	66.15	84.63	64.58	81.30	60.87	84.18	60.52	81.38	32.63	53.97
	He	41.65	72.55	21.05	48.37	51.40	77.68	42.50	72.75	48.35	76.18	11.35	10.02
	EOC	59.83	82.92	64.58	82.97	64.52	83.73	62.30	84.05	66.57	82.48	59.60	71.57
	Proposed	**68.23**	**86.75**	**66.53**	**87.13**	**67.63**	**86.82**	**66.38**	**86.23**	**69.51**	**86.43**	**65.65**	**84.02**
FMNIST	Xavier	67.65	74.53	68.92	76.13	66.75	73.78	67.10	72.65	66.85	74.47	49.38	69.88
	He	61.13	74.60	44.60	65.25	62.92	76.75	62.88	76.55	64.97	75.22	11.50	11.70
	EOC	67.22	76.87	66.85	73.83	66.93	76.45	67.00	77.63	67.08	75.67	65.48	73.07
	Proposed	**70.67**	**77.43**	**70.63**	**78.17**	**71.55**	**77.92**	**68.62**	**77.80**	**68.17**	**78.33**	**67.93**	**76.43**

Experiments with Activations $\omega \approx 1$ (Network Size)

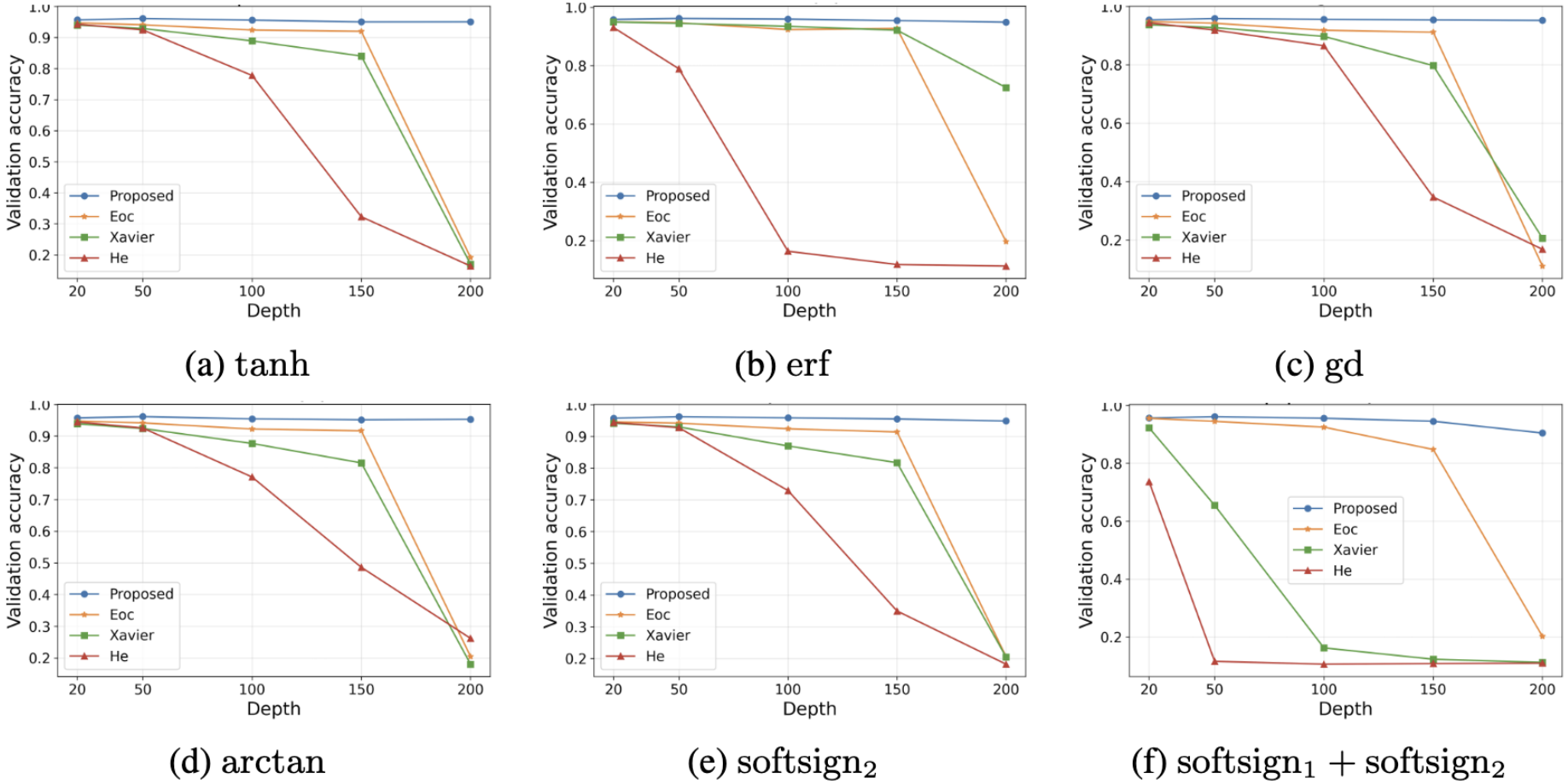


Figure 12: MNIST validation accuracy versus depth for FFNNs (width 64) with odd-sigmoid activations and four initializations (Proposed, EOC, Xavier, He). Each panel fixes one activation and shows the best validation accuracy over 10 epochs for depths $L \in \{20, 50, 100, 150, 200\}$.

Experiments with Activations $\omega \approx 1$ (Network Size)

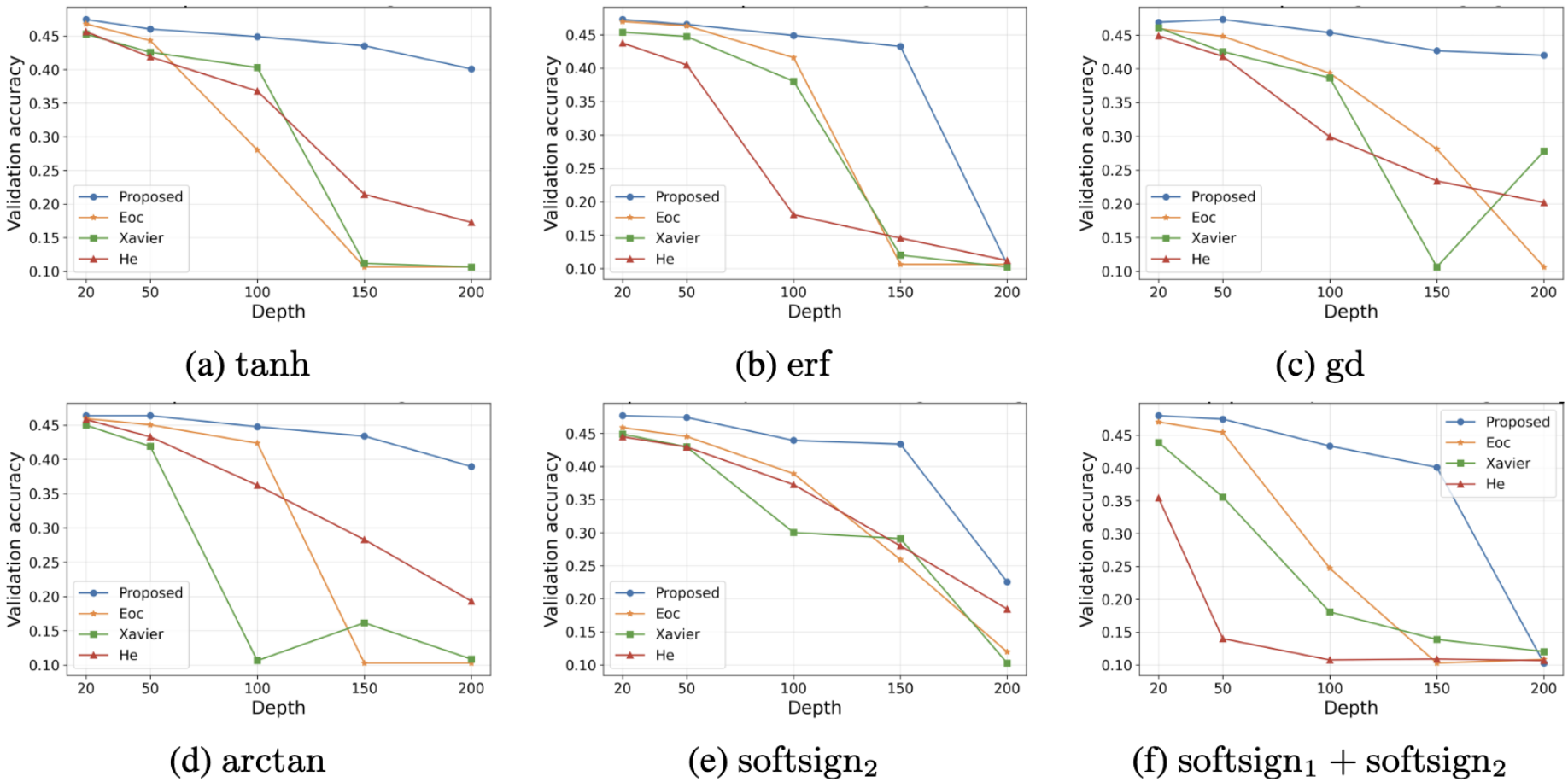
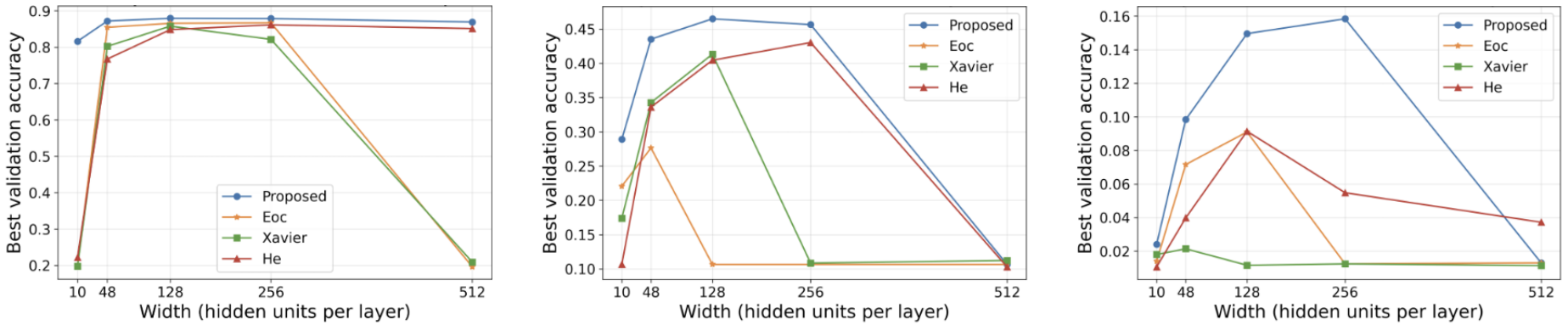


Figure 13: CIFAR 10 validation accuracy versus depth for FFNNs (width 64) with odd-sigmoid activations and four initializations (Proposed, EOC, Xavier, He). Each panel fixes one activation and shows the best validation accuracy over 10 epochs for depths $L \in \{20, 50, 100, 150, 200\}$.

Experiments with Activations $\omega \approx 1$ (Network Size)



(a) Fashion MNIST (b) CIFAR 10 (c) CIFAR 100

Figure 14: Best validation accuracy versus width $\{10, 48, 128, 150, 200, 512\}$ for a 100 layer tanh FFNN. Each curve shows the Proposed, EOC, Xavier, and He initialization schemes, and each point corresponds to the best validation accuracy over 20 training epochs.

Experiments with Activations $\omega \not\approx 1$

Corollary 1.

Let $f_1, f_2 \in \mathcal{F}$ and let $c_1, c_2 \geq 0$ with $(c_1, c_2) \neq (0, 0)$. If $g = c_1 f_1 + c_2 f_2$, then $g \in \mathcal{F}$. Furthermore, it holds that

$$\frac{1}{\omega_g} = \frac{c_1}{\omega_{f_1}} + \frac{c_2}{\omega_{f_2}}.$$

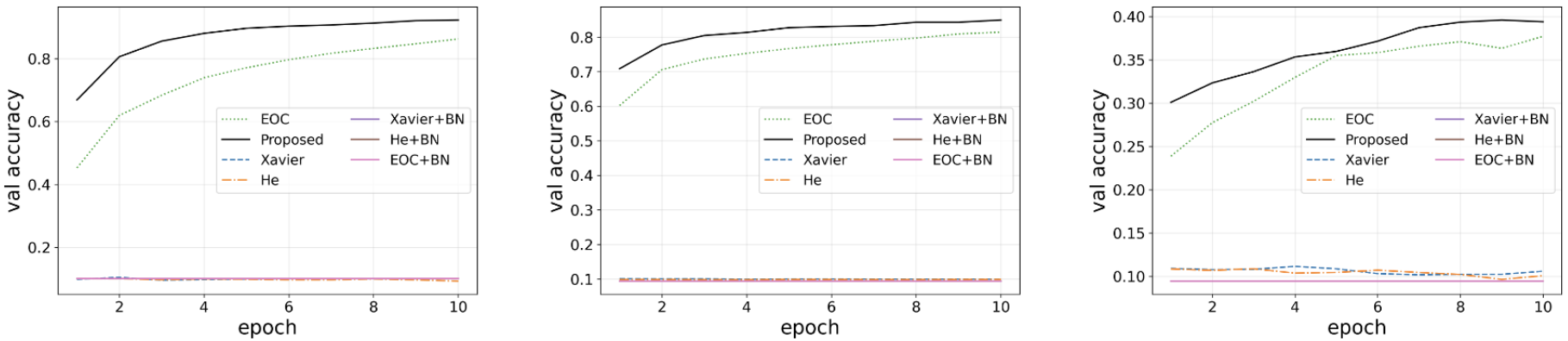
Since $\mathcal{F}$ is closed under addition, more generally, any finite sum $\sum_{j=1}^{M} c_j f_j \in \mathcal{F}$ for all $c_j \geq 0$ with $(c_1, \ldots, c_M) \neq \mathbf{0}$ and $f_j \in \mathcal{F}$.

For

$$f(x) = \tanh(ax) + \operatorname{erf}(bx) + \frac{x}{1 + |cx|} + \operatorname{gd}(dx), \tag{1}$$

by Corollary this f is an odd–sigmoid activation $\omega = 1/(a + \frac{2}{\sqrt{\pi}} b + c + d)$.

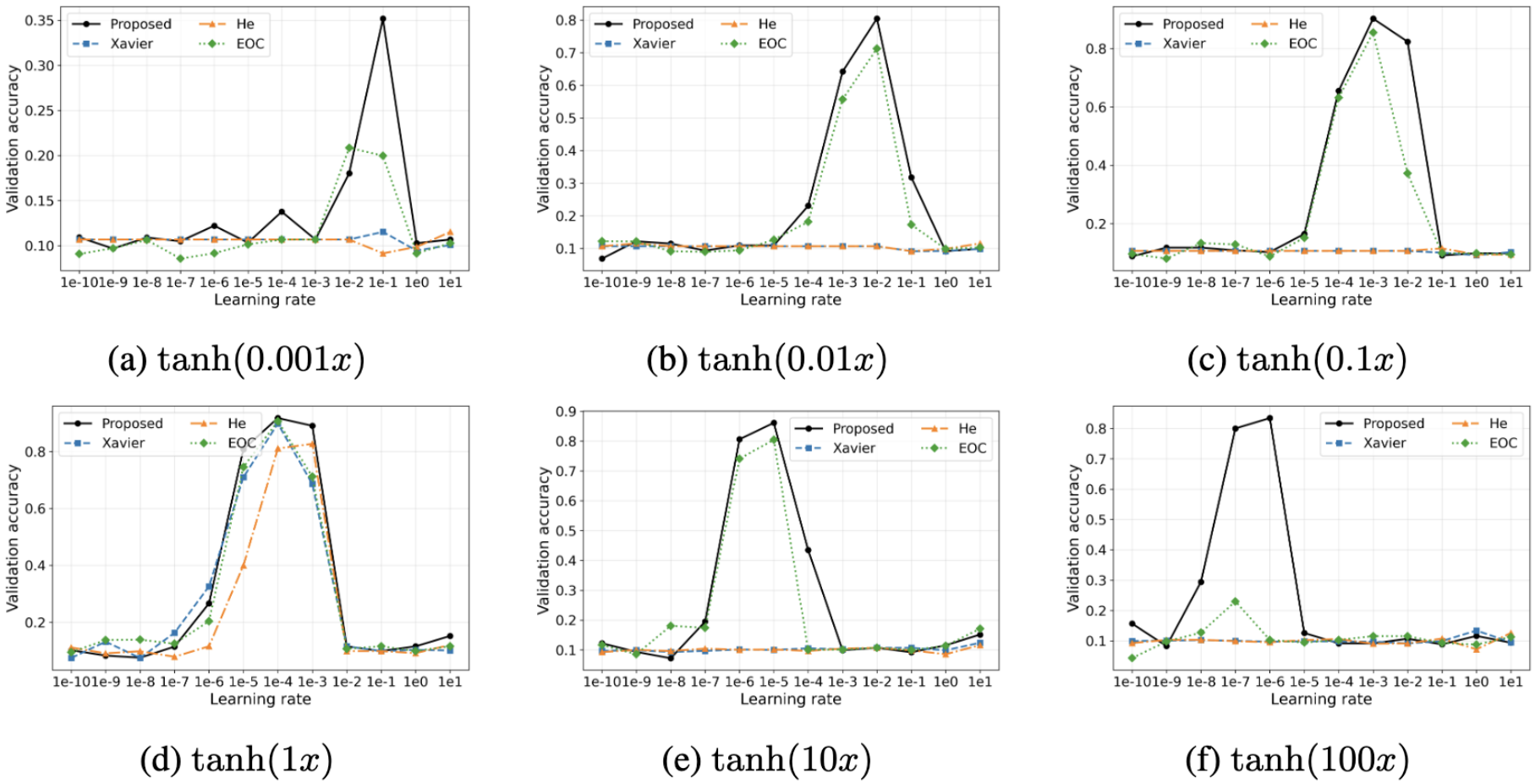
Experiments with Activations $\omega \not\approx 1$ (Normalization)



(a) MNIST (b) FMNIST (c) CIFAR10

Figure 15: Val Acc for the $f(x) = \tanh(ax) + \mathrm{erf}(bx) + x/(1 + |cx|) + \mathrm{gd}(dx)$ in FFNN with 100 hidden layers of width 64. We compare seven strategies: Proposed, Xavier, He, EOC, and their BN variants. Each panel uses a different dataset and a different choice of coefficients (a, b, c, d): **(a)** (10,1000,10,1), **(b)** (100,1000,10,1000)), **(c)**(1000,100,0.1,0.01).

Experiments with Activations $\omega \not\approx 1$ (Learning rate)

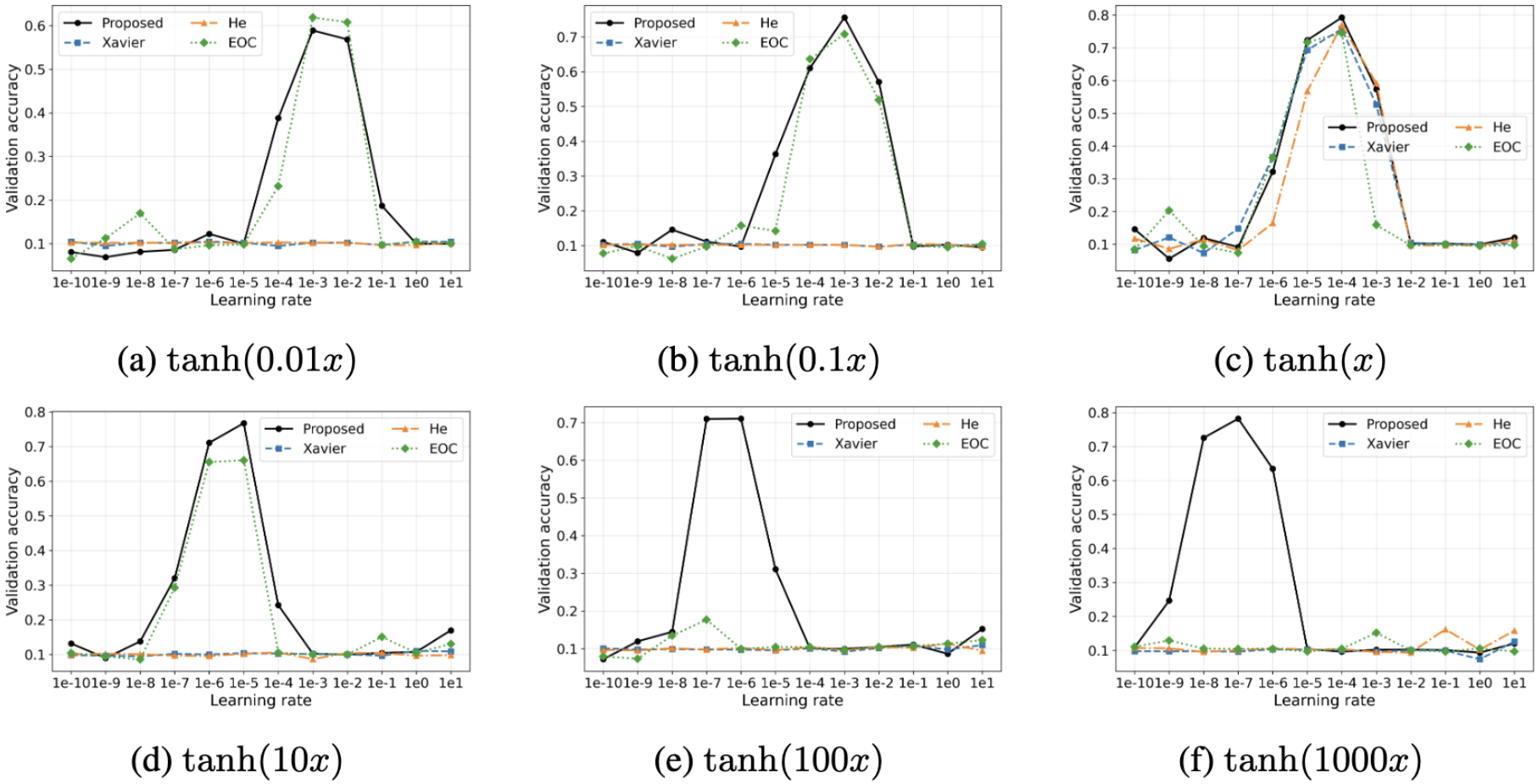


(a) $\tanh(0.001x)$ (b) $\tanh(0.01x)$ (c) $\tanh(0.1x)$

(d) $\tanh(1x)$ (e) $\tanh(10x)$ (f) $\tanh(100x)$

Figure 16: Learning rate accuracy curves on MNIST for a 20 layer, width 512 feedforward network with activation $f(x) = \tanh(\alpha x)$. Each panel corresponds to a different activation scale $\alpha \in \{10^2, 10^1, 1, 10^{-1}, 10^{-2}, 10^{-3}\}$. For each learning rate, we train for 200 iterations on a 1k training subset.

Experiments with Activations $\omega \not\approx 1$ (Learning rate)



(a) $\tanh(0.01x)$ (b) $\tanh(0.1x)$ (c) $\tanh(x)$

(d) $\tanh(10x)$ (e) $\tanh(100x)$ (f) $\tanh(1000x)$

Figure 17: Learning rate accuracy curves on Fashion MNIST for a 20 layer, width 512 feedforward network with activation $f(x) = \tanh(\alpha x)$. Each panel corresponds to a different activation scale $\alpha \in \{10^2, 10^1, 1, 10^{-1}, 10^{-2}, 10^{-3}\}$. For each learning rate, we train for 200 iterations on a 1k training subset.

Experiments with Activations $\omega \not\approx 1$ (Scale)

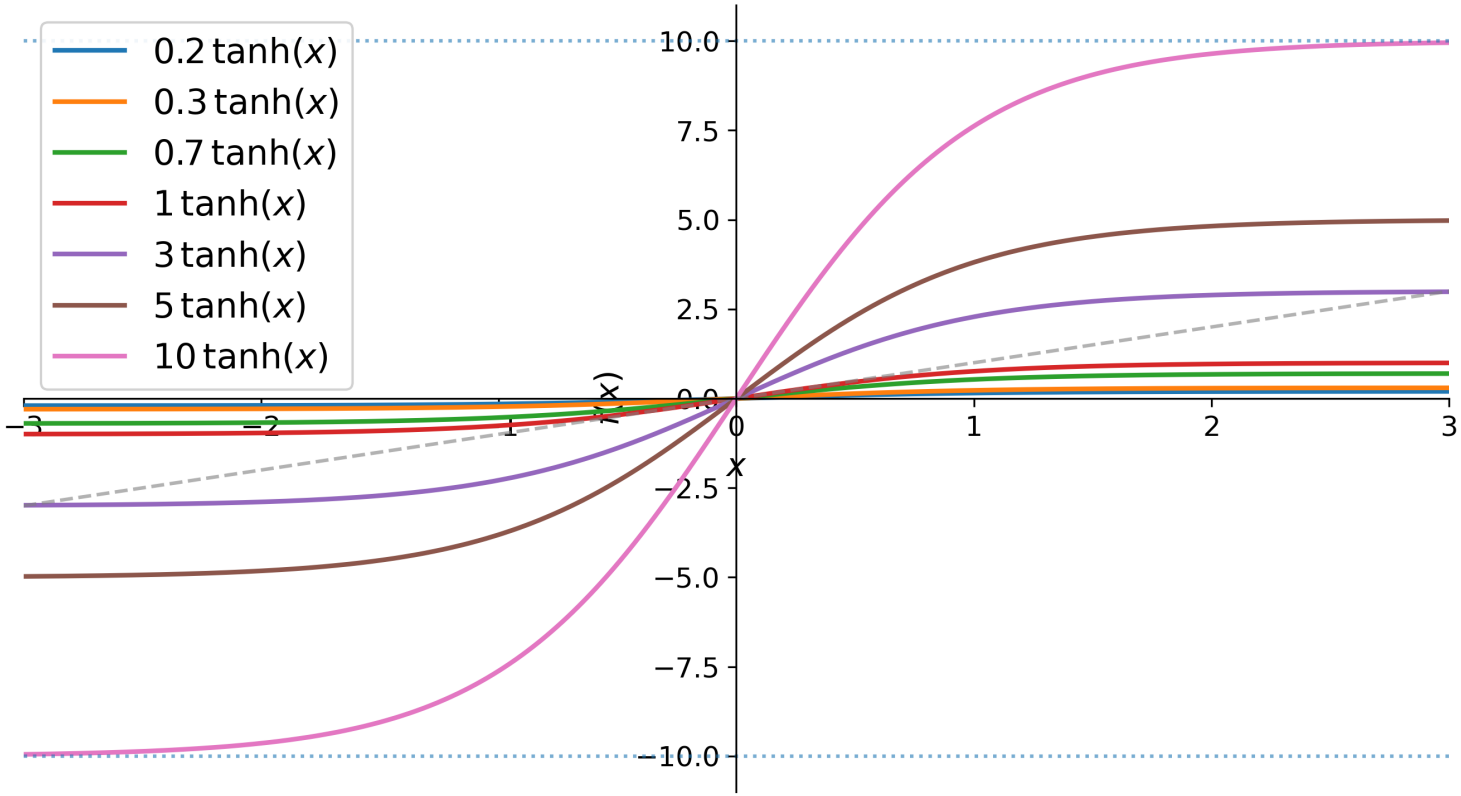


Figure 18: Examples of odd-sigmoid function.

Physics-Informed Neural Networks

A **Physics-Informed Neural Network (PINN)** [19] integrates physical laws, such as PDEs, into the training process to ensure the model adheres to these constraints. This approach allows PINNs to solve scientific problems efficiently, even with limited data.

Total Loss Function

The total loss $\mathcal{L}_{\text{total}}$ is a weighted sum of all components:

$$\mathcal{L}_{\text{total}} = \lambda_1 \mathcal{L}_{\text{data}} + \lambda_2 \mathcal{L}_{\text{physics}} + \lambda_3 \mathcal{L}_{\text{boundary}} + \lambda_4 \mathcal{L}_{\text{initial}}$$

where $\lambda_1, \lambda_2, \lambda_3, \lambda_4$ are weights to balance each term.

Experiments with Activations $\omega \not\approx 1$ (Scale-PINN)

Black–Scholes Equation

$$\frac{\partial V}{\partial t} + \tfrac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} + rS\frac{\partial V}{\partial S} - rV = 0, \qquad S \in [0,1],\ t \in [0,1].$$

with the initial condition

$$V(S,0) = \max(S-K,0), \qquad K = 0.5,$$

and parameters $\sigma = 0.2,\ r = 0.05.$

Experiments with Activations $\omega \not\approx 1$ (Scale-PINN)

Question.

How can we improve line fitting performance in vanilla physics-informed neural networks?

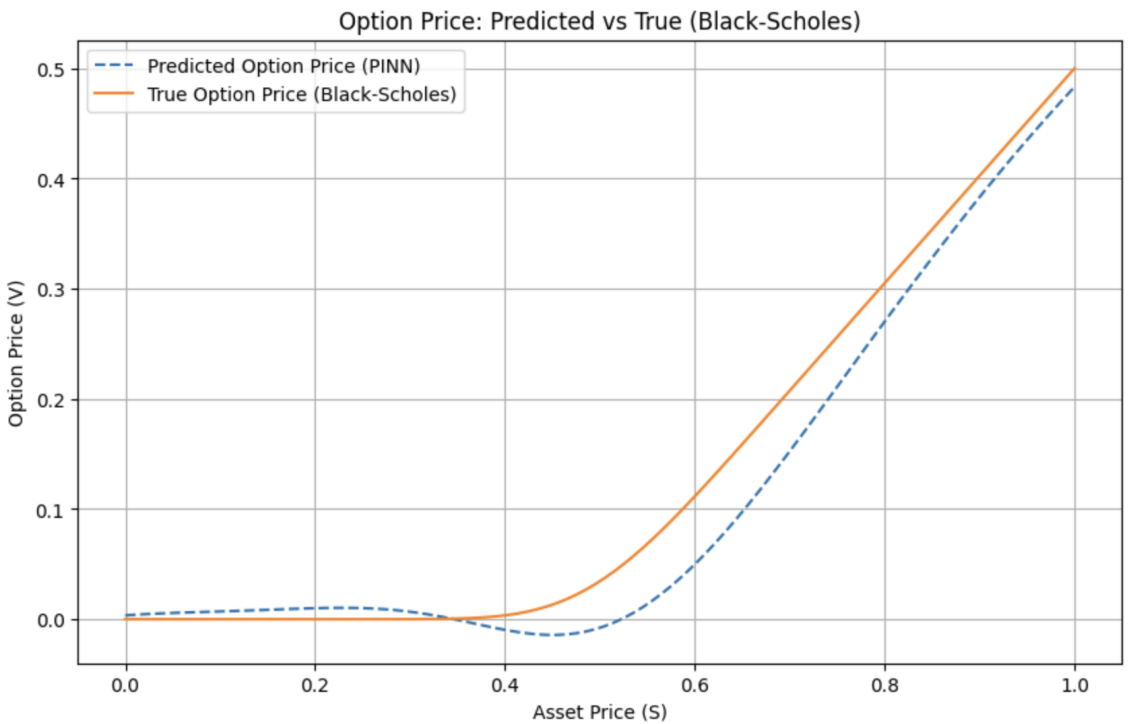


Figure 19: Predicted vs True

Experiments with Activations $\omega \not\approx 1$ (Scale-PINN)

Question.

Would adjusting the range of the activation distribution help with line fitting?

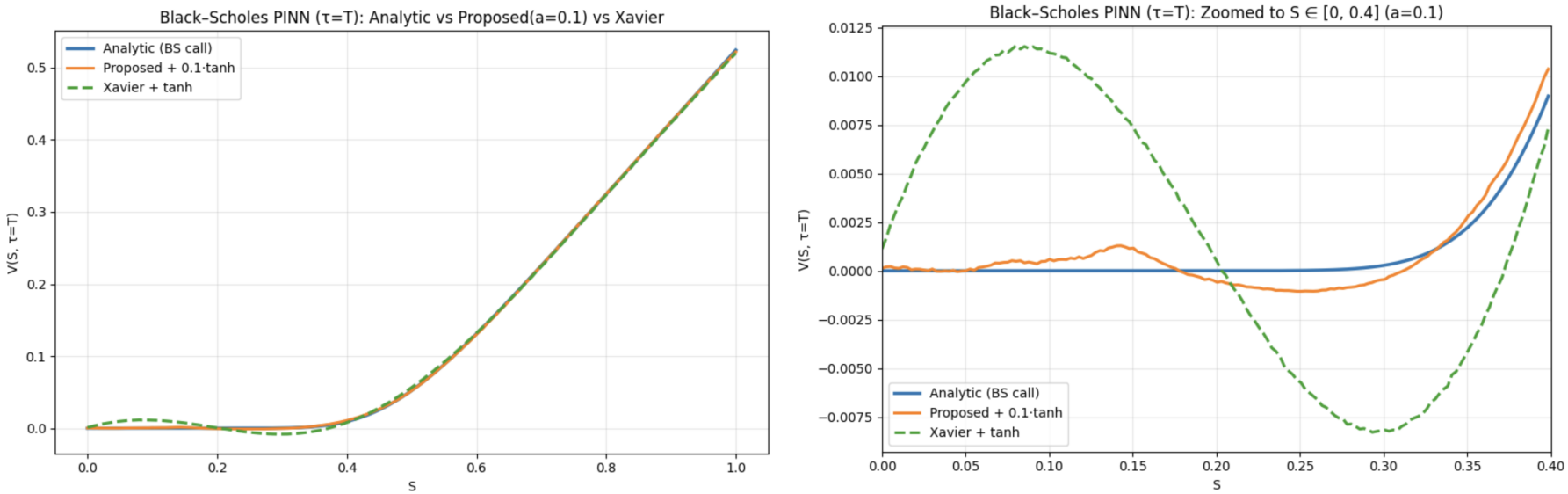


Figure 20: 0.1tanh(x)

With this smaller scale, the PINN approximation matches the exact solution better than with the standard $\tanh(x)$.

Experiments with Activations $\omega \not\approx 1$ (Scale-PINN)

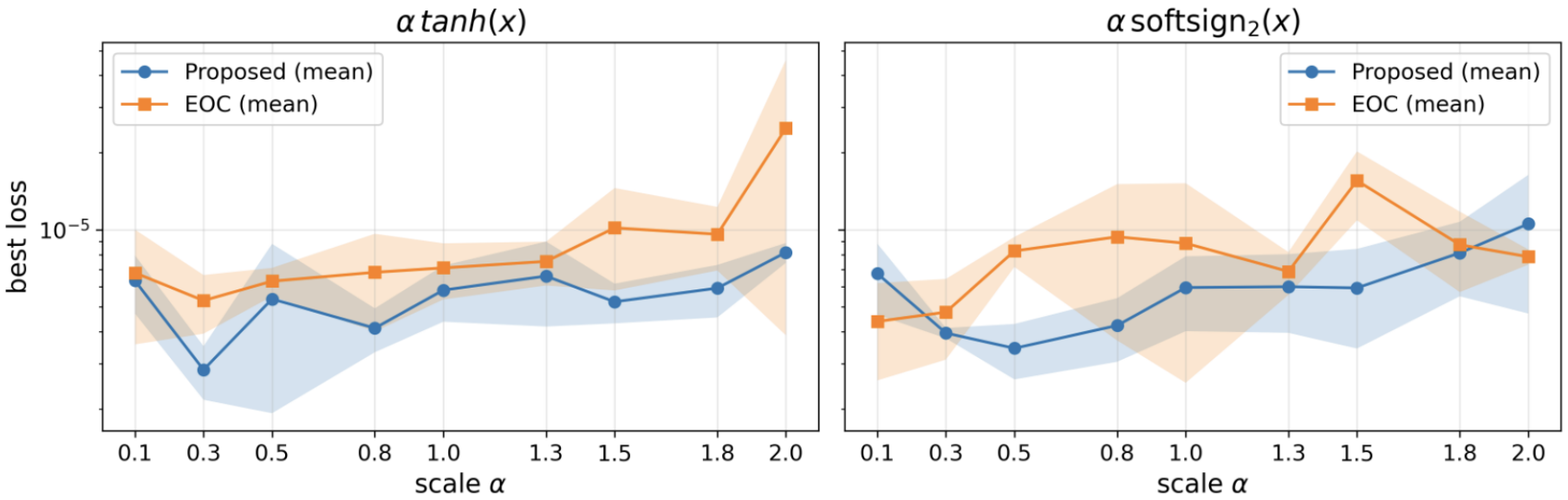


Figure 38: Best PINN loss versus activation scale for the Burgers PINN (depth 50, width 64), comparing the proposed and EOC initializations. Left: $a\tanh(x)$ with $a \in \{0.1, 0.3, 0.5, 0.8, 1.0, 1.3, 1.5, 1.8, 2.0\}$. Right: $b\ \mathrm{softsign}_2(x)$ with the same set of scales.

Experiments with Activations $\omega \not\approx 1$ (Scale-PINN)

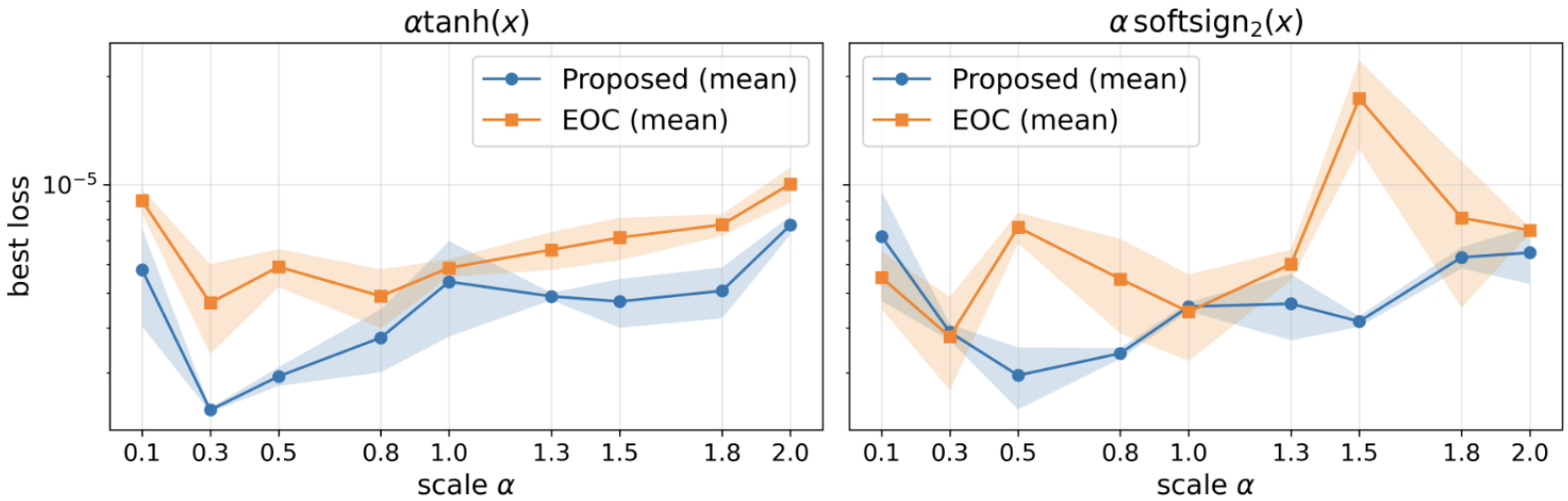
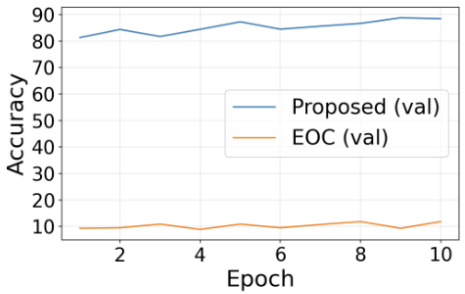
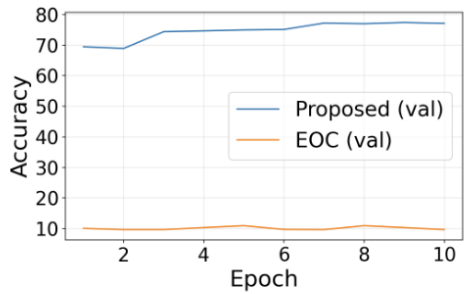
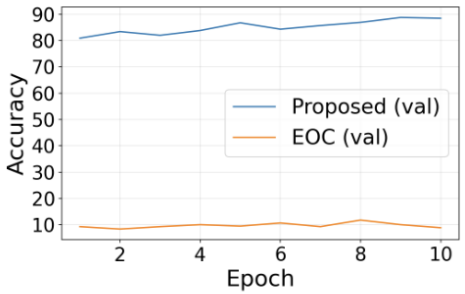
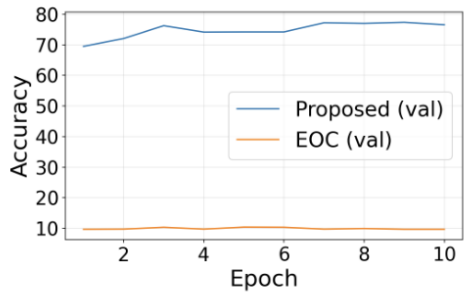
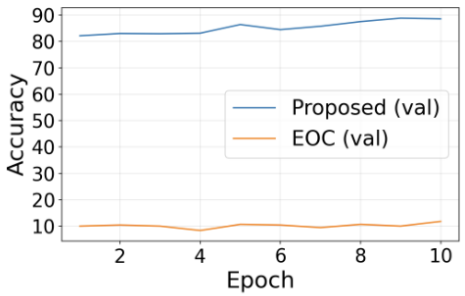
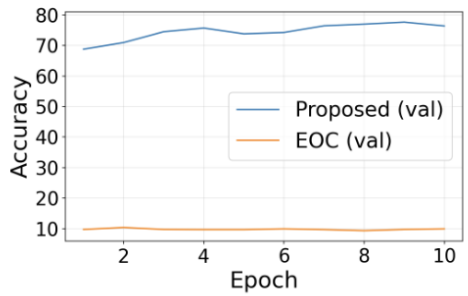


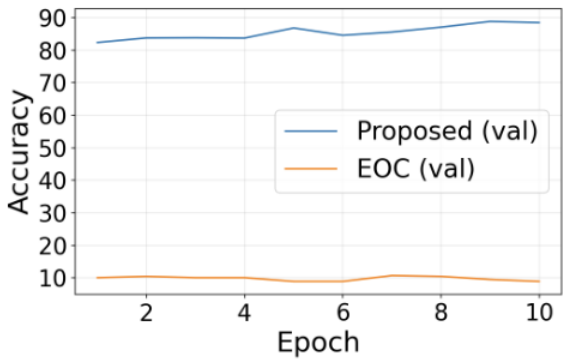
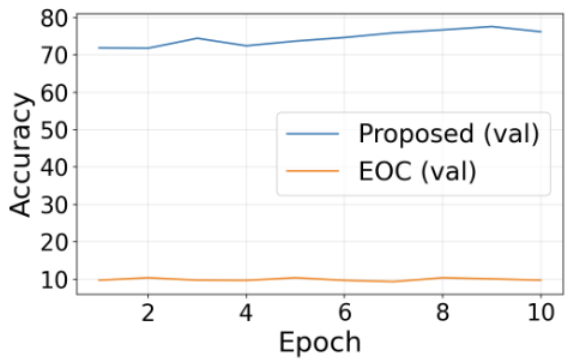
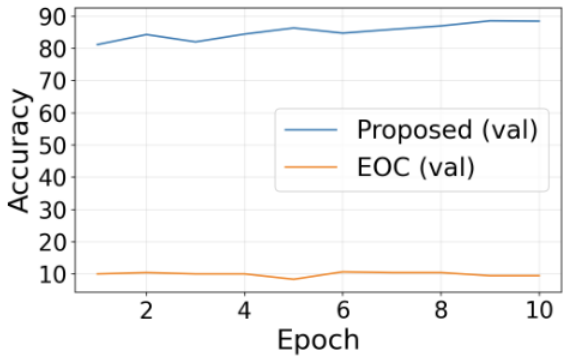
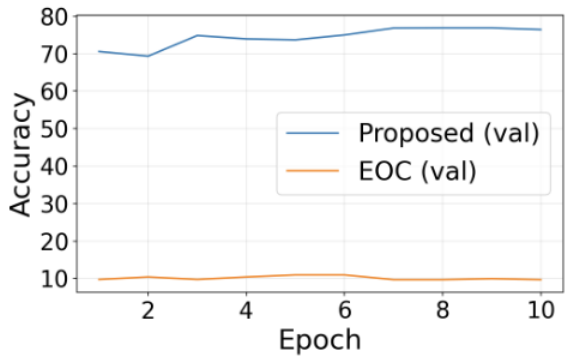
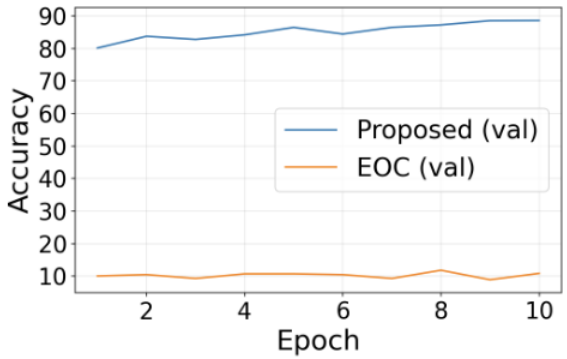
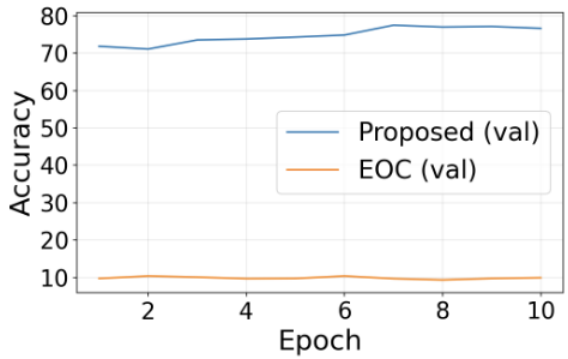
Figure 7: Best PINN loss versus activation scale for the Black–Scholes PINN (depth 50, width 64), comparing the proposed and EOC initializations. **(Left)** $a\tanh(x)$ with $a \in \{0.1, 0.3, 0.5, 0.8, 1.0, 1.3, 1.5, 1.8, 2.0\}$. **(Right)** $b\ \mathrm{softsign}_2(x)$ with the same set of scales.

Experiments with Activations $\omega \not\approx 1$

Table 3: Validation accuracy on MNIST (left) and Fashion MNIST (right) for a 50 layer, width 128 fully connected neural network with activation $a\tanh(x)$. Each row corresponds to a different activation scale a, and for every a the learning rate is set to $\eta = 10^{-4}/a$ for both initializations.

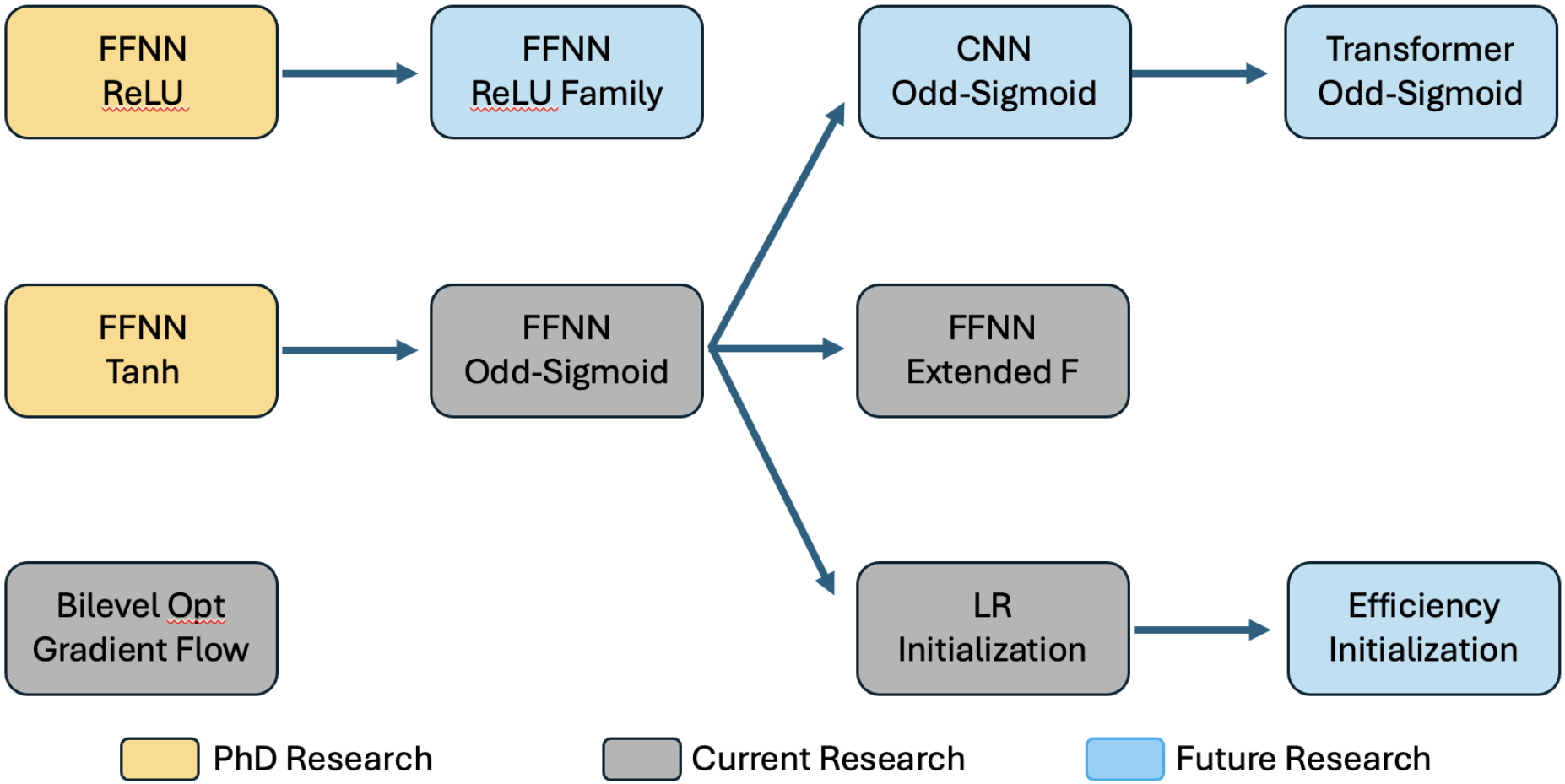
a	MNIST (accuracy vs. epoch)	Fashion-MNIST (accuracy vs. epoch)
10^4		
10^5		
10^6		

Experiments with Activations $\omega \not\approx 1$

10^7		
10^8		
10^9		

5. Future Work

Future Work


FFNN
ReLU
FFNN
ReLU Family
CNN
Odd-Sigmoid
Transformer
Odd-Sigmoid
FFNN
Tanh
FFNN
Odd-Sigmoid
FFNN
Extended F
Bilevel Opt
Gradient Flow
LR
Initialization
Efficiency
Initialization
PhD Research
Current Research
Future Research

References I

[1] Narkhede, Meenal V., Prashant P. Bartakke, and Mukul S. Sutaone. (2022) A review on weight initialization strategies for neural networks." Artificial intelligence review 55.1: 291-322.

[2] Kumar, Siddharth Krishna. (2017) On weight initialization in deep neural networks. arXiv preprint arXiv:1704.08863 (2017).

[3] He, J., Li, L., Xu, J., & Zheng, C. (2018). ReLU deep neural networks and linear finite elements. arXiv preprint arXiv:1807.03973.

[4] Hanin, B., & Sellke, M. Approximating continuous functions by ReLU nets of minimal width (2018). arXiv preprint arXiv:1710.11278.

[5] Yarotsky, D. (2017). Error bounds for approximations with deep ReLU networks. Neural Networks, 94, 103-114.

References II

[6] A. T. Puig, A. Wiesel, G. Fleury, and A. O. Hero, "Multidimensional shrinkage-thresholding operator and group LASSO penalties," IEEE Signal Process. Lett., vol. 18, no. 6, pp. 363–366, Jun. 2011.

[7] Apicella, A., Donnarumma, F., Isgrò, F., & Prevete, R. (2021). A survey on modern trainable activation functions. Neural Networks, 138, 14-32.

[8] Ba, J. L., Kiros, J. R., & Hinton, G. E. (2016). Layer normalization. arXiv preprint arXiv:1607.06450.

[9] Glorot, X., & Bengio, Y. (2010, March). 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.

References III

[10] Lu, L., Shin, Y., Su, Y., & Karniadakis, G. E. (2019). Dying relu and initialization: Theory and numerical examples. arXiv preprint arXiv:1903.06733.

[11] Burkholz, R., & Dubatovka, A. (2019). Initialization of relus for dynamical isometry. Advances in Neural Information Processing Systems, 32.

[12] He, K., Zhang, X., Ren, S., & Sun, J. (2015). Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In Proceedings of the IEEE international conference on computer vision (pp. 1026-1034).

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

References IV

[14] Zhao, J., Schäfer, F., & Anandkumar, A. (2021). Zero initialization: Initializing neural networks with only zeros and ones. arXiv preprint arXiv:2110.12661.

[15] Lee, H., Kim, Y., Yang, S., & Choi, H. (2024). Improved weight initialization for deep and narrow feedforward neural network. arXiv preprint arXiv:2311.03733.

[16] Rathore, Pratik, et al. (2024) Challenges in training PINNs: A loss landscape perspective. ICML2024.

[17] Poole, Ben, et al. (2016) Exponential expressivity in deep neural networks through transient chaos." Advances in neural information processing systems 29.

[18] Raghu, Maithra, et al. (2017) On the expressive power of deep neural networks." International conference on machine learning. PMLR.

References V

[19] Karniadakis, George Em, et al. (2021) Physics-informed machine learning. Nature Reviews Physics 3.6.

[20] Lee, H., Choi, H., & Kim, H. (2024). Robust Weight Initialization for Tanh Neural Networks with Fixed Point Analysis. arXiv preprint arXiv:2410.02242.

[21] Zhang, Shijun, Jianfeng Lu, and Hongkai Zhao. "Deep network approximation: Beyond relu to diverse activation functions." Journal of Machine Learning Research 25.35 (2024): 1-39.

[22] Bingham, Garrett, and Risto Miikkulainen. "Efficient activation function optimization through surrogate modeling." Advances in Neural Information Processing Systems 36 (2023): 6634-6661.

References VI

[23] Lee, Hyunwoo, Hayoung Choi, and Hyunju Kim. "Signal Preserving Weight Initialization for Odd-Sigmoid Activations." arXiv preprint arXiv:2509.23085 (2025).

[24] Schoenholz, Samuel S., et al. "Deep information propagation." arXiv preprint arXiv:1611.01232 (2016).

[25] Hayou, Soufiane, Arnaud Doucet, and Judith Rousseau. "On the impact of the activation function on deep neural networks training." International conference on machine learning. PMLR, 2019.

Thank you for your attention!