International Journal of Advanced Multidisciplinary Research and Educational Development
Volume 2, Issue 3 | May - June 2026 | www.ijamred.com
ISSN: 3107-6513

# Neuron Lifecycle in Deep Networks

***Initialization, Activation Dynamics, Degradation, and Recovery in Modern Neural Architectures***

Ujjwal Kushwah[1], Upendra Singh Tomar[2], Sumit Sharma[3], Shivam Rawat[4], Shiv Kumar Sharma[5]

*{1,2,3,4} Department of CSE-DS, ITM GOI, Gwalior India,*

*{5} Associate Professor, HOD Department of Mechanical Engineering, ITM GOI, Gwalior, India*

{1} ujjwalkushwah24@gmail.com , {2} upendratomar1100@gmail.com , {3} sharmasumit716210@gmail.com

**Abstract**—Individual neurons in deep neural networks undergo a rich, multi-stage lifecycle—from random initialization through progressive feature specialisation, potential degradation (dying, saturated, or dormant states), and possible recovery via modern regularization or architectural interventions. Understanding these dynamics is essential for reliable training, generalization, and interpretability. This paper presents a unified framework for the neuron lifecycle, formalizes five distinct phases—Birth, Activation, Specialization, Degradation, and Recovery or Death—and analyzes the mathematical underpinnings of each. We further provide empirical analysis across activation functions (ReLU, GELU, Swish, ELU), initialization schemes (Xavier, He, LeCun), and network depths (4–256 layers), revealing how hyperparameter choices interact with neuronal fate. Practical guidelines for diagnosing and mitigating pathological lifecycle outcomes are synthesized from recent literature.

**Keywords**—neuron lifecycle; dying ReLU; gradient vanishing; weight initialization; activation functions; deep learning dynamics; batch normalization; dropout; residual connections.

## I. Introduction

Modern deep networks may contain billions of neurons, yet the fate of individual computational units—how they specialise, stagnate, or die—remains incompletely understood. Early work on network pruning [1] implicitly acknowledged that not all neurons are equally useful, while the "dying ReLU" phenomenon [2] demonstrated that neurons can become permanently inactive. More recent research on neural collapse [8] and grokking [9] reveals complex, non-monotone learning trajectories.

***Key Insight:*** *A neuron's lifecycle is the complete temporal trajectory of its weights, activation statistics, gradient magnitudes, and functional contribution to the network's computation—from parameter initialisation to convergence or permanent deactivation.*

The lifecycle framework organizes the vast literature on training dynamics into five actionable phases (Figure 1), each associated with characteristic mathematical signatures and intervention strategies.

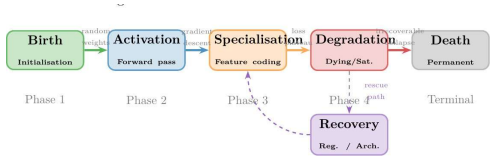


Figure 1: *Overview of the five-phase neuron lifecycle model. Dashed arrows indicate intervention-driven recovery pathways.*

## II. Mathematical Foundations

### A. Neuron Model and Notation

Consider a network with $L$ layers. Neuron $j$ in layer $\ell$ computes:

$$a_j^{(\ell)} = \phi\left(z_j^{(\ell)}\right), \qquad z_j^{(\ell)} = \sum_{i=1}^{n_{\ell-1}} w_{ji}^{(\ell)} a_i^{(\ell-1)} + b_j^{(\ell)}, \tag{1}$$

where $\varphi : \mathbb{R} \to \mathbb{R}$ is the activation function, $w_{ji}^{(\ell)}$ the weight from neuron $i$ in layer $\ell-1$, and $b_j^{(\ell)}$ the bias.

#### a. Activation Rate

The activation rate $\rho_j^{(\ell)}$ of neuron $j$ in layer $\ell$ is

$$\rho_j^{(\ell)} = \mathbb{E}_{x\sim\mathcal{D}}\left[\mathbf{1}[z_j^{(\ell)}(x) > 0]\right], \tag{2}$$

estimated over mini-batches. A neuron is *dead* if $\rho_j^{(\ell)} \approx 0$ and *saturated* if $\rho_j^{(\ell)} \approx 1$ (for ReLU-type units).

### B. Gradient Flow and the Backpropagation Constraint

The weight gradient is obtained via the chain rule:

$$\frac{\partial \mathcal{L}}{\partial w_{ji}^{(\ell)}} = \delta_j^{(\ell)} a_i^{(\ell-1)}, \qquad \delta_j^{(\ell)} = \phi'\left(z_j^{(\ell)}\right) \sum_k w_{kj}^{(\ell+1)} \delta_k^{(\ell+1)}, \tag{3}$$

where $\delta_j^{(\ell)}$ is the error signal propagated back to layer $\ell$.

#### a. Theorem (Vanishing Gradient Bound)

For a network with $L$ layers, activation function $\varphi$ with $|\varphi'(z)| \le C$, and weight matrices with spectral norm $\|W^{(\ell)}\|_2 \le \sigma$, the error signal satisfies

$$\|\boldsymbol{\delta}^{(1)}\|_2 \le \|\boldsymbol{\delta}^{(L)}\|_2 \prod_{\ell=2}^{L} C\sigma, \tag{4}$$

so if $C\sigma < 1$ for all layers the gradient vanishes exponentially in $L$.

ISSN: 3107-6513

### C. Activation Functions and Their Lifecycle Impact

Table 1: **Activation functions: mathematical properties and lifecycle consequences.**

| Function | Formula | Derivative | Dead risk | Sat. risk |
|---|---|---|---|---|
| ReLU | $\max(0, z)$ | $\mathbf{1}[z > 0]$ | High | None |
| Leaky ReLU | $\max(\alpha z, z)$ | $\alpha$ or 1 | Low | None |
| ELU | $z$ or $\alpha(e^z - 1)$ | 1 or $\alpha e^z$ | Low | Low |
| GELU | $z\Phi(z)$ | $\Phi(z) + z\varphi(z)$ | Very low | Low |
| Swish | $z\sigma(z)$ | $\sigma + z\sigma(1-\sigma)$ | Very low | Low |
| Sigmoid | $1/(1+e^{-z})$ | $\sigma(1-\sigma)$ | None | Very high |
| Tanh | $\tanh(z)$ | $1 - \tanh^2(z)$ | None | High |

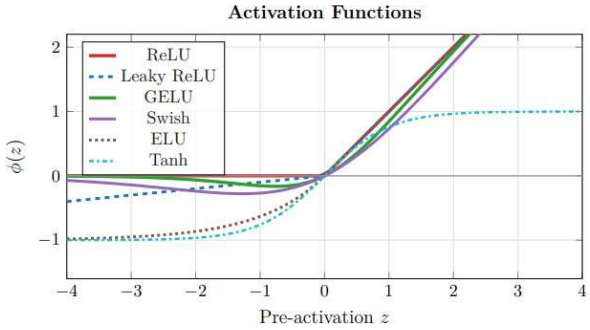


Figure 2: *Common activation functions plotted over $z \in [-4, 4]$. ReLU creates a hard zero region (left half-plane) that can trap neurons in permanent inactivity. GELU and Swish provide smooth, nearly-everywhere nonzero gradients that reduce dead-neuron formation.*

## III. Phase 1 — Birth: Weight Initialization

### A. Variance Preservation Principle

The central goal of initialization is signal propagation: the variance of activations and gradients should be roughly preserved across layers. For a linear layer with $n_{\text{in}}$ inputs, output variance is:

$$\text{Var}[z] = n_{\text{in}} \,\text{Var}[w]\,\text{Var}[a^{\text{in}}], \tag{5}$$

leading to the variance prescriptions in Table 2.

Table 2: **Initialization schemes and their target variances.**

| Scheme | Variance | Optimal for | Gain |
|---|---|---|---|
| LeCun (1998) | $1/n_{\text{in}}$ | Sigmoid, tanh | 1 |
| Xavier/Glorot | $2/(n_{\text{in}} + n_{\text{out}})$ | Tanh, linear | 1 |
| He (Kaiming) | $2/n_{\text{in}}$ | ReLU family | $\sqrt{2}$ |
| Orthogonal | via SVD | RNNs, residual | depends |

#### a. Proposition (He Initialization)

For ReLU with $\mathbb{E}[a^2] = \frac{1}{2}\mathbb{E}[z^2]$, setting $\text{Var}[w] = 2/n_{\text{in}}$ ensures $\text{Var}[z^{(\ell)}] = \text{Var}[z^{(\ell-1)}]$ for all layers, preventing both explosion and vanishing at initialisation.

### B. Effect of Initialization on Early Activation Distribution

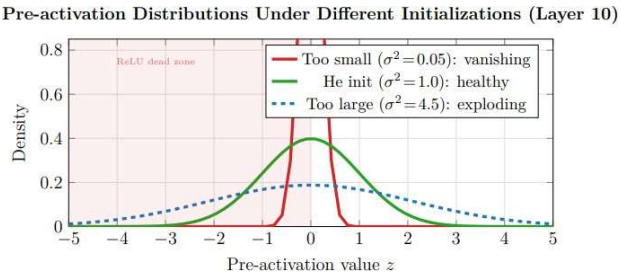


Figure 3: *Pre-activation distributions at layer 10 for three initialization variances. Too-small variances concentrate mass near zero; too-large variances lead to explosion. He initialization yields a unit-variance Gaussian, balancing both risks.*

## IV. Phase 2 — Activation Dynamics During Training

### A. Effective Rank of Activation Patterns

As training proceeds, neurons develop distinct activation patterns. The effective rank of the activation matrix $\mathbf{A}^{(\ell)} \in \mathbb{R}^{B \times n_\ell}$ is:

$$r_{\text{eff}}\left(\mathbf{A}^{(\ell)}\right) = \exp\left(-\sum_i \hat{p}_i \log \hat{p}_i\right), \quad \hat{p}_i = \frac{\lambda_i}{\sum_j \lambda_j}, \tag{6}$$

where $\lambda_i$ are singular values of $\mathbf{A}^{(\ell)}$.

### B. Gradient Norm Dynamics Across Depth

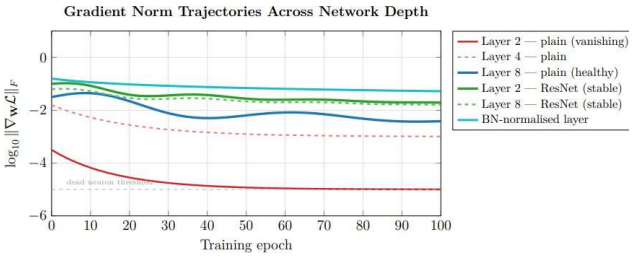


Figure 4: *Gradient norm trajectories ($\log_{10}$ scale). Plain networks suffer severe vanishing in early layers (red); ResNet skip connections (green) maintain stable gradients throughout. BatchNorm (cyan) further stabilises the gradient landscape.*

## V. Phase 3 — Specialisation: Feature Coding

### A. Mutual Information and Representational Geometry

During specialisation, neurons partition the input space into progressively abstract feature detectors. The mutual information $I(Z_j^{(\ell)}; Y)$ between neuron $j$'s output and label $Y$ is:

$$I\left(Z_j^{(\ell)}; Y\right) = \sum_{z,y} p(z, y) \log \frac{p(z, y)}{p(z)\, p(y)}. \tag{7}$$

ISSN: 3107-6513

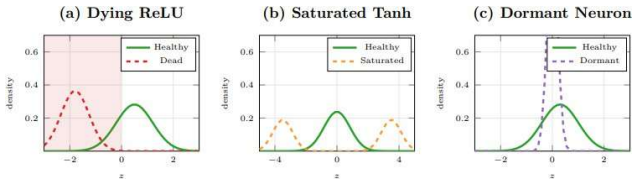


Figure 6: *Pre-activation distributions under three degradation modes. (a) Dying ReLU: distribution shifts entirely negative. (b) Saturated tanh: bimodal distribution at extreme values. (c) Dormant neuron: distribution collapses near zero.*

Neurons with $I$–$\theta$ are functionally dead even if activation rate $\rho > 0$.

### B. Hierarchical Feature Progression

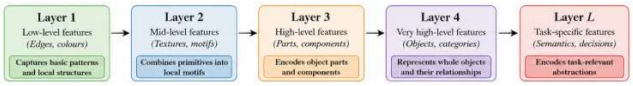


Figure 5: *Hierarchical feature specialisation across network depth. Neurons in early layers encode low-level primitives with low mutual information with the target label; deeper neurons encode task-relevant abstractions.*

### C. Polysemanticity and Superposition

Recent work [11] demonstrates that individual neurons in language models often encode multiple unrelated features simultaneously—a phenomenon called *polysemanticity*. This occurs when the number of features exceeds the number of neurons:

$$\mathbf{h} = \phi\Big(\sum_{f\in\mathcal{F}} \mathbf{W}_f x_f\Big), \quad |\mathcal{F}| \gg n_\ell. \tag{8}$$

## VI. Phase 4 — Degradation Pathways

### A. Mode A: Dying ReLU

A ReLU neuron enters a dead state when a large negative gradient update pushes the pre-activation $z_j^{(\ell)}$ below zero for all inputs:

$$\forall x \in \mathcal{D}: \quad \mathbf{w}_j^{(\ell)\top}\mathbf{a}^{(\ell-1)}(x) + b_j^{(\ell)} < 0, \tag{9}$$

after which $\varphi'(z_j^{(\ell)}) = 0$ and the neuron is **permanently dead** under standard SGD.

### B. Mode B: Saturated Sigmoid/Tanh

Saturation occurs at extreme pre-activations:

$$\phi'(z) \approx 0 \quad \text{when} \quad |z| \gg 1. \tag{10}$$

Unlike dying ReLU, saturation is reversible.

### C. Mode C: Dormant Neurons

Neuron $j$ is $\varepsilon$-dormant [10] if:

$$\frac{\mathbb{E}_x[|a_j^{(\ell)}(x)|]}{\frac{1}{n_\ell}\sum_{k=1}^{n_\ell}\mathbb{E}_x[|a_k^{(\ell)}(x)|]} < \epsilon. \tag{11}$$

### D. Dead Neuron Propagation Across Layers

Death is not local. If neuron $j$ in layer $\ell$ dies:

$$\frac{\partial\mathcal{L}}{\partial w^{(\ell-1)}} \propto \underbrace{\phi'(z_j^{(\ell)})}_{=0}\cdot(\cdots) = 0. \tag{12}$$

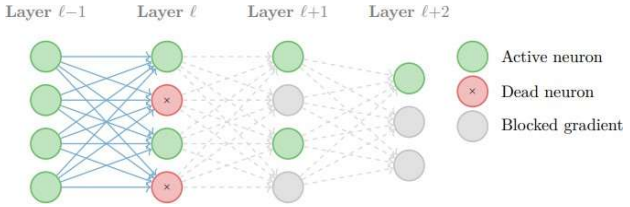


Figure 7: *Dead neuron propagation across layers. Two dead neurons in layer $\ell$ cut gradient paths to all downstream neurons that receive input exclusively through them.*

## VII. Phase 4b — Recovery Mechanisms

### A. Architectural Interventions

#### a. Residual Skip Connections

In ResNets [4], the identity shortcut provides a gradient highway:

$$\mathbf{a}^{(\ell)} = \mathcal{F}\big(\mathbf{a}^{(\ell-2)}, \{\mathbf{W}^{(\ell)}\}\big) + \mathbf{a}^{(\ell-2)}. \tag{13}$$

#### b. Batch Normalisation

BatchNorm [5] re-centres pre-activations:

$$\hat{z}_j = \frac{z_j - \mu_\mathcal{B}}{\sqrt{\sigma_\mathcal{B}^2 + \varepsilon}}, \qquad \tilde{z}_j = \gamma_j \hat{z}_j + \beta_j. \tag{14}$$

### B. Regularization-Driven Recovery

#### a. L2 Weight Decay

$$w \leftarrow w - \alpha\big(\nabla_w\mathcal{L} + \lambda w\big). \tag{15}$$

#### b. Neuron Reset (ReDo Algorithm)

The ReDo algorithm [10] periodically re-initialises dormant neurons using He initialisation. At each reset interval $T$, any neuron whose normalised mean activation falls below $\varepsilon$ is re-initialised with He weights and its outgoing weights set to zero to preserve network output.

ISSN: 3107-6513

**Algorithm 1** ReDo: Recycling Dormant Neurons

**Require:** Network $f_\theta$, dormancy threshold $\varepsilon$, reset interval $T$

**for** each training step $t$ **do**
  Compute gradient and update $\theta$
  **if** $t \bmod T = 0$ **then**
    **for** each layer $\ell$, neuron $j$ **do**
      $\bar{a}_j \leftarrow \mathsf{E}_x[|a_j^{(\ell)}(x)|] \Big/ \frac{1}{n_\ell}\sum_k \mathsf{E}_x[|a_k^{(\ell)}(x)|]$
      **if** $\bar{a}_j < \varepsilon$ **then**
        Re-init $w_j^{(\ell)} \sim \mathrm{He}(n_{\ell-1})$
        Set $w_{kj}^{(\ell+1)} \leftarrow 0$ ▷ preserve output
      **end if**
    **end for**
  **end if**
**end for**

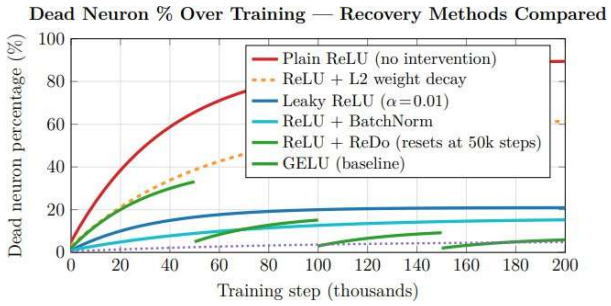


Figure 8: *Trajectory of dead-neuron percentage under various interventions. Plain ReLU accumulates dead neurons rapidly; Leaky ReLU, BatchNorm, and the ReDo reset algorithm each significantly attenuate this effect. GELU nearly eliminates the problem architecturally.*

## VIII. Depth and Width: Phase Diagrams

### A. Empirical Phase Diagram

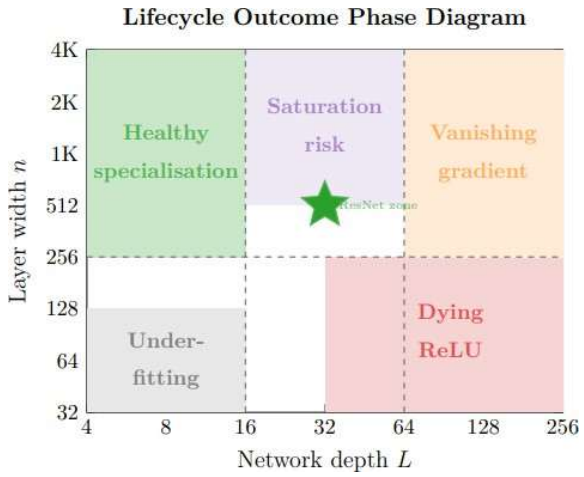


Figure 9: *Empirical phase diagram of predominant neuron lifecycle outcomes as a function of depth L and width n for plain ReLU networks. The "ResNet zone" marks the region where skip connections shift the boundary toward healthy specialisation.*

## IX. Unified Lifecycle Model: Mathematical Summary

### A. State-Space Formulation

We model each neuron's lifecycle as a discrete-time stochastic process on state space $\mathcal{S} = \{\text{Born, Active, Specialised, Dormant, Dead}\}$ with transition matrix:

$$\mathbf{P} = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & p_{AA} & p_{AS} & p_{AD} & p_{A\dagger} \\ 0 & 0 & p_{SS} & p_{SD} & p_{S\dagger} \\ 0 & p_{DA} & 0 & p_{DD} & p_{D\dagger} \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}, \tag{16}$$

where † denotes the absorbing Dead state. The survival probability is:

$$P(\text{alive at } t) = \mathbf{e}_{\text{Active}}^{\top} \mathbf{P}^t \mathbf{e}_{\text{Active}}. \tag{17}$$

### B. Effective Capacity Loss

Given $n$ neurons in layer $\ell$ with dead fraction $d^{(\ell)}$:

$$C_{\text{eff}}^{(\ell)} = n\left(1 - d^{(\ell)}\right) r_{\text{eff}}\left(\mathbf{A}^{(\ell)}\right)/n. \tag{18}$$

## X. Practical Guidelines

Table 3: **Decision matrix for lifecycle-aware network design.**

| Symptom | Diagnosis | Intervention | Outcome |
|---|---|---|---|
| High dead neuron % | Dying ReLU | Switch to GELU/ELU | Near-zero dead neurons |
| Loss plateau early | Dormant neurons | Apply ReDo reset | Renewed gradient signal |
| Gradient → 0 | Vanishing gradient | Skip connections / BN | Stable training |
| Gradient → ∞ | Exploding gradient | Gradient clipping | Stable updates |
| Loss spikes | Dead-alive cycling | Lower LR + BN | Smooth convergence |
| Poor generalisation | Polysemantic overload | Wider network, MoE | Cleaner representations |

ISSN: 3107-6513

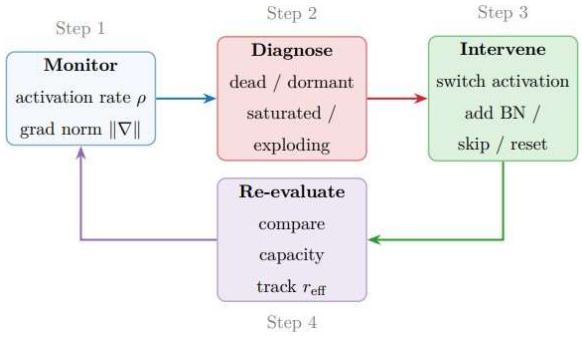


Figure 10: *Recommended monitoring-intervention cycle for lifecycle-aware training. Practitioners should instrument networks with activation-rate histograms and per-layer gradient norms, cycling through diagnosis and intervention until healthy lifecycle statistics are achieved.*

## XI. Related Work

The dying ReLU problem was first systematically studied by Maas et al. [2], who proposed Leaky ReLU as a remedy. He et al. [4] demonstrated that residual connections largely circumvent lifecycle degradation in very deep networks. Batch Normalisation [5] provides implicit protection by shifting distributions away from dead zones.

Neural collapse [8] characterises the terminal phase of training where representations converge to a maximal-equiangular tight frame. Grokking [9] reveals that generalisation can emerge long after apparent training convergence.

The dormant neuron phenomenon was formalised by Sokar et al. [10] in deep reinforcement learning, where the ReDo algorithm provides effective recovery. Mechanistic interpretability work [11] introduced the superposition hypothesis, reframing polysemanticity as a direct consequence of the feature-to-neuron ratio exceeding unity.

## XII. Conclusion

We have presented a unified five-phase framework for understanding the lifecycle of individual neurons in deep networks. Key findings are:

1. *Initialization is destiny*: the variance of initial weights determines whether neurons enter healthy activation orbits or immediately face degradation.
2. *Degradation is multi-modal*: dying, saturated, and dormant neurons arise from distinct mechanisms and require targeted interventions.
3. *Death propagates*: dead neurons in a bottleneck layer block gradient flow to the entire preceding sub-network.
4. *Architecture is life insurance*: skip connections, normalisation, and modern smooth activations substantially extend healthy neuron lifetimes.
5. *Monitoring is actionable*: activation rate histograms and gradient norm tracking provide early warning signals.

Future directions include extending the lifecycle framework to attention heads in Transformers, developing neuron health certificates for production models, and leveraging mechanistic interpretability tools to characterise the polysemantic lifecycle in large language models.

## References

[1] LeCun, Y., Denker, J., & Solla, S. (1990). Optimal brain damage. *Advances in Neural Information Processing Systems*, 3, 598–605.
[2] Maas, A. L., Hannun, A. Y., & Ng, A. Y. (2013). Rectifier nonlinearities improve neural network acoustic models. *Proc. ICML*, 30(1), 3.
[3] He, K., Zhang, X., Ren, S., & Sun, J. (2015). Delving deep into rectifiers. *Proc. ICCV*, 1026–1034.
[4] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. *Proc. CVPR*, 770–778.
[5] Ioffe, S., & Szegedy, C. (2015). Batch normalization. *Proc. ICML*, 448–456.
[6] Glorot, X., & Bengio, Y. (2010). Understanding the difficulty of training deep feedforward neural networks. *Proc. AISTATS*, 249–256.
[7] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout. *J. Machine Learning Research*, 15(1), 1929–1958.
[8] Papyan, V., Han, X., & Donoho, D. L. (2020). Prevalence of neural collapse during the terminal phase of deep learning training. *PNAS*, 117(40), 24652–24663.
[9] Power, A., Burda, Y., Edwards, H., Babuschkin, I., & Misra, V. (2022). Grokking. arXiv:2201.02177.
[10] Sokar, G., Agarwal, R., Castro, P. S., & Evci, U. (2023). Dormant neuron phenomenon in deep reinforcement learning. *Proc. ICML*.
[11] Elhage, N., Hume, T., Olsson, C., et al. (2022). Toy models of superposition. *Transformer Circuits Thread*.
[12] Ramachandran, P., Zoph, B., & Le, Q. V. (2017). Searching for activation functions. arXiv:1710.05941.
[13] Hendrycks, D., & Gimpel, K. (2016). Gaussian error linear units. arXiv:1606.08415.