Communications on Applied Nonlinear Analysis
ISSN: 1074-133X
Vol 32 No. 8s (2025)

Mitigating Neural Network Training Challenges Through Effective Optimizers and Activation Function Selection

Syed Shouib Ahmed[1], Ramadevi KS[2], Rukhsar[3], Reshma S[4], Anusha S[5]

[1]*HOD of Computer Science, Krupanidhi College of Commerce and Management.*

[2,3,4]*Assistant Professor, Krupanidhi College of Commerce and Management.*

[5]*Assistant Professor, Sri Jagadguru RenukacharyaCollege of Science Arts and Commerce.*

Article History:

Received:10-07-2025

Revised:27-08-2025

Accepted:10-09-2025

Abstract:
Training deep neural networks is often hindered by problems such as vanishing gradients (Sigmoid), neuronal instability (ReLU), slow convergence (SGDM), and poor generalization (Adam). This research systematically investigates how the choice of optimizers and activation functions mitigates these challenges. Using aConvolutional Neural Network(CNN)trainedon the MNISTdataset, five activation functions (Sigmoid, ReLU, LeakyReLU, GeLU, SiLU) and four optimizers (SGDM, Adam, AdamW, RMSProp) were compared in 20 unique configurations. Performance was evaluated on the basis of validation accuracy, convergence speed, and gradient stability.

Keywords: Deep Learning, Convolutional Neural Networks (CNN), Activation Functions, Optimizers, Gradient Vanishing Problem, Convergence Speed, MNIST Dataset

1. Introduction

Deep learning models, particularly Convolutional Neural Networks, have revolutionized the fields of computer vision, speech recognition, and natural language processing. Despite their success, the training process of deep networks remains sensitive to the choice of activation functions and optimization algorithms. Improper selection often results in issues such as vanishing or exploding gradients, neuronal dead zones, and unstable convergence. During backpropagation, the gradients of the loss function with respect to the weights determine the learning dynamics. Mathematically, this process is governed by the following rule:

$$w_{t+1} = w_t - \eta \frac{\partial \mathcal{L}}{\partial w_t} \quad (1)$$

Where w_t denotes the model parameters at time step t,η represents the learning rate, and $\mathscr{L}$ is the loss function. If the gradient approaches zero (vanishing gradient) or becomes excessively large (exploding gradient), the model either fails to learn or diverges during training.

Furthermore, activation functions such as the Sigmoid and ReLU families significantly affect the gradient flow. For example, the Sigmoid function

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

2

tends to squash inputs into a narrow range (0,1), leading to gradient saturation, while the ReLU function $f(x) = max(0, x)$ (3)

suffers from the dying neuron problem where neurons permanently output zero for negative inputs.

Optimizers like Adam, RMS Prop, and Adam W attempt to stabilize the update process by adaptively scaling learning rates and maintaining moving averages of gradients. However, their interplay with activation functions in determining training efficiency and gradient stability remains underexplored.

This study aims to systematically analyze how various combinations of activation functions and optimizers impact the convergence behavior, training stability, and gradient flow of CNN architectures.

2. OBJECTIVES

- Analyze the effect of different activation functions on model convergence, stability, and neuron activity.
- Evaluate how various optimizers influence gradient flow, weight updates, and learning speed.
- Identify the most effective activation-optimizer pair for stable and efficient CNN training on bench-mark dataset (MNIST).
- Provide a quantitative comparison using validation accuracy, gradient norms, and convergence curves..

3. LiteratureReview

Optimization forms a central challenge in training deep neural networks, which involves navigating com- plex, non-convex loss functions across high-dimensional spaces. The foundation of this process is gradient descent, implemented efficiently via backpropagation [1] [2], which uses first-order information to itera- tively update weight parameters towards an optimal solution [1]. Achieving training stability is paramount to ensure efficient convergence and favorable generalization [3]. Stability issues often arise due to inherent optimization challenges such as the accumulation of noise in gradient estimates (particularly with small batch sizes), or pathologies related to gradient flow, notably vanishing or exploding gradients [1] [2]. Furthermore, optimization techniques influence the structure of the loss landscape explored; empirical evidence suggests that minimizing generalization

error often correlates with converging to wider basins of attraction, rather than sharp minima [1]. The evolution of Activation Functions (AFs) primarily focuses on mitigating these stability issues, particularly the vanishing gradient problem prevalent in older saturating functions like the Logistic Sigmoid [4] [5]. The introduction of the Rectified Linear Unit (ReLU) [1] [6] provided computational speed benefits (as it avoids calculating exponentials) and promoted sparsity. However, ReLU suffers from the "dying ReLU" effect where neurons output zero gradients indefinitely for negative inputs [2]. To address this, the Leaky ReLU (LReLU) was proposed [2] [4] [6], assigning a small, non-zero slope (αx) to negative inputs, thereby keeping gradients alive during training [2]. Further advancements aimed at integrating smoothness and nonmonotonicity: the Sigmoid-weighted Linear Unit (SiLU), also known as Swish [4] [3] [6], is a self-gated function known for smooth, non-monotonic properties [2]. Similarly, the Gaussian Error Linear Unit (GELU) [4] [3] [6], derived from a probabilistic framework, is a smooth and non-monotonic alternative reported to perform well across tasks [4] [6]. Optimization algorithms have also advanced significantly to improve efficiency. While Stochastic Gradient Descent with Momentum (SGDM) leverages past updates to ac celerate convergence and reduce oscillation [3], adaptive learning rate methods provide dynamic adjustments. RMSProp utilizes a running aver- age of the squared gradient magnitudes to normalize the learning rate adaptively [3] [7]. Adam [7] [1] [5] combines both momentum (v) and adaptive learning rates (r) [1] [3]. Despite Adam's rapid convergence, standard application of L2 regularization (weight decay) often leads to suboptimal generalization performance compared to SGDM [8]. This drawback motivated the introduction of AdamW [4] [7], which explicitly decouples weight decay from the adaptive gradient up dates, often resulting in robust performance 4 and improved generalization. Comparative studies on image classification tasks show AdamW yielding superior accuracy compared to standard Adam, RMSProp, and SGDM [8]. While activation functions and optimizers are funda mental components of modern deep learning, the interplay between these two design choices remains inadequately explored in a systematic manner across the existing literature [3] [2]. Current studies often focus separately on surveying numerous AFs [4] [6] or comparing optimizers [8] [3], but network performance fundamentally depends on their complex interacttion [2]. This thesis addresses the consequential research gap by conducting a rigorous, combined performance evaluation of modern activation functions (ReLU, LReLU, GELU, SiLU) and high-performance adaptive optimizers (SGDM, RMSProp, Adam, AdamW)[9]. Through systematic experimentation, this work identifies synergistic combinations that enhance training stability, accelerate convergence,

and improve generalization, thereby contributing necessary empirical evidence for effective architectural design [10]. The synergistic potential arising from the specific selection of Activation Functions (AFs) and optimization algorithms remains an underexplored dimension of deep learning research [11], despite the crit icality of both components in achieving stable and generalizable model training [12]. Consequently, this thesis presents a systematic comparative analysis of state-of-the-art AFs paired with modern adaptive optimizers to derive empirically validated guidelines for neural network architecture design

4. MODELARCHITECTURE

The proposed model, referred to as Simple CNN, was designed to evaluate how activation functions and optimizersinfluenceconvergenceandgradientflow.Thenetworkconsists of two convolutional blocks followed by fully connected layers, as shown below:

$$\text{Input}(1\times 28\times 28)\rightarrow \begin{Bmatrix}\text{Conv}(1,32,3)\\ \text{Activation}\\ \text{MaxPool}(2)\end{Bmatrix}\rightarrow \begin{Bmatrix}\text{Conv}(32,64,3)\\ \text{Activation}\\ \text{MaxPool}(2)\end{Bmatrix}\rightarrow \text{FC}(3136,128)\rightarrow\text{Activation}\rightarrow \text{FC}(128,10)$$

Each convolutional layer utilized a 3×3 kernel with padding of 1 to preserve spatial dimensions. Max-poolinglayerswitha2×2 kernelwereusedtodownsamplethefeaturemaps.Thefullyconnectedlayers transformed the extracted features into class logits.

5. EXPERIMENTALSETUPANDWORKDONE

All experiments were performed using PyTorch on GPU mode with manual seeding to ensure reproducibility. The computational environment included Python 3.10, Torch 2.x, and Matplotlib for visualization. Five activation functions were tested - Sigmoid, ReLU, LeakyReLU, GELU, and SiLU across four optimizers: SGD with momentum (SGDM), Adam, AdamW, and RMSProp. Each combination of activation-optimizer pair was trained for five epochs, resulting in a total of 20 unique experiments. During training, loss, accuracy, and average gradient norms were recorded after every epoch.

The gradient norm metric was included to analyze the smoothness and stability of weight updates under different activation and optimizer settings.

6. QUANTITATIVEMETRICSFORANALYSIS

Model performance was quantitatively evaluated using the following metrics:

- Training and Validation Accuracy: To measure learning effectiveness and generalization ability.
- Cross-Entropy Loss: The categorical loss function used for multi-class classification.
- Gradient Norms: The average L2 norm of gradients across all trainable parameters, representing the magnitude of updates and the presence of vanishing or exploding gradients.

7. MATHEMATICALFORMULATION

For each input image $x_i$with label y_i, the Convolutional Neural Network (CNN) computes a probability distribution over the output classes as:

$$y\hat{}i = Softmax(f\theta(xi))$$

Where $f0$ represents the network parameterized by weights 0.

The optimization objective minimizes the categorical cross-entropy loss N

$$\mathcal{L} = -\frac{1}{N}\sum_{i=1}^{N} y_i \log(\hat{y}_i)$$

The weight update rule varies depending on the choice of optimizer:

- **StochasticGradientDescentwithMomentum(SGDM):**

$$vt = \beta vt - 1 + (1-\beta)\nabla\theta\mathcal{L}, \theta t + 1 = \theta t - \eta vt$$

Where β denotes the momentum co efficient and η is the learning rate.

- **RMS Prop:**

$$E[g^2]_t = \beta E[g^2]_{t-1} + (1-\beta)g_t^2$$
$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{E[g^2]_t + \epsilon}} g_t$$

where $E[g^2]_t$is the exponentially weighted moving average of squared gradients. RMS Prop adaptively scales learning rates, reducing oscillations in the optimization path.

- **Adam:**

$$m_t = \beta_1 m_{t-1} + (1-\beta_1)\nabla_\theta \mathcal{L}, v_t = \beta_2 v_{t-1} + (1-\beta_2)(\nabla_\theta \mathcal{L})^2$$
$$\theta_{t+1} = \theta_t - \eta \frac{m_t/(1-\beta_1^t)}{\sqrt{v_t/(1-\beta_2^t)} + \epsilon}$$

Where m_t and v_t are the first and second moment estimates of the gradient.

- **Adam W (Decoupled Weight Decay):**

$$m_t = \beta_1 m_{t-1} + (1-\beta_1)\nabla_\theta \mathcal{L}$$
$$v_t = \beta_2 v_{t-1} + (1-\beta_2)(\nabla_\theta \mathcal{L})^2$$
$$\theta_{t+1} = \theta_t - \eta \left(\frac{m_t/(1-\beta_1^t)}{\sqrt{v_t/(1-\beta_2^t)} + \epsilon} + \lambda \theta_t \right)$$

where λ denotes the weight decay coefficient. AdamW improves generalization by decoupling weightdecay from the gradient update.

The gradient norm was computed at each epoch to monitor the scale of parameter updates:

$$\|\nabla_\theta\|_2 = \sqrt{\sum_i (\nabla_{\theta_i})^2}$$

A stable and moderate gradient norm indicates effective learning and prevents exploding or vanishinggradients during training.

8. ACTIVATION FUNCTIONS

Thestudycomparedfiveactivationfunctions,eachcontributingdifferentlytolearningstabilityand expressiveness:

- Sigmoid:$f(x)=\frac{1}{1+e^{-x}}$, compresses out puts to (0,1), prone to vanishing gradients
- ReLU: $f(x)$=max(0,x),introduces sparsity and mitigates vanishing gradientissues.
- Leaky ReLU: $f(x)$ = max(αx,x), where α = 0.01, prevents dead neurons by allowing small gradients for negative inputs.
- GELU:$f(x)=x\Phi(x)$,combines ReLU and drop out characteristics, providing smoother activation.
- SiLU(Swish): $f(x)=x\cdot$ Sigmoid(x),offers self-gating behavior and improved gradient flow.

9. RESULTS

The experiments revealed that the choice of activation function and optimizer significantly impacts trainingstability,convergence,andmodelaccuracy.Across20uniquetrainingruns, a clear hierarchy of component effectiveness emerged on the MNIST dataset.

Table1: Maximum Validation Accuracy Comparison across Optimizers and Activation Functions (MNIST)

Activation	SGDM	RMSProp	Adam	AdamW
Sigmoid	0.3241	0.1009	0.9803	0.9790
ReLU	0.9862	0.9901	0.9906	0.9872
LeakyReLU	0.9847	0.9903	0.9910	0.9885
GELU	0.9866	0.9918	0.9895	0.9901
SiLU	0.9873	0.9907	0.9905	0.9893

Table2: Average Gradient Norm Comparison across Optimizers and Activation Functions (MNIST)

Activation	SGDM	RMSProp	Adam	AdamW
Sigmoid	0.312	0.677	0.358	0.339
ReLU	1.00	0.545	0.747	0.835
LeakyReLU	1.04	0.463	0.614	0.664
GELU	0.99	0.507	0.528	0.582
SiLU	1.01	0.450	0.589	0.542

Gradient norms exhibited consistent patterns across all runs. The Sigmoid function, particularly under SGDM and RMS Prop, produced extremely low gradient magnitudes (≈0.3−0.6), confirming vanishing gradients and poor learning. In contrast,$ReLU$ –based functions maintained higher, stable Grad Norms

(≈ 0.6−1.0), indicating strong gradient propagation. Adaptive optimizers (Adam, AdamW, RMS Prop) consistently reduced Grad Norm across epochs while achieving higher accuracies, demonstrating effective gradient normalization and stability.

Figure 1: Performance Comparison: All Activations with Adam (Left) and AdamW (Right). The plots show the superior convergence speed of ReLU-variants over Sigmoid and the subtle performance differences between the two adaptive optimizers.

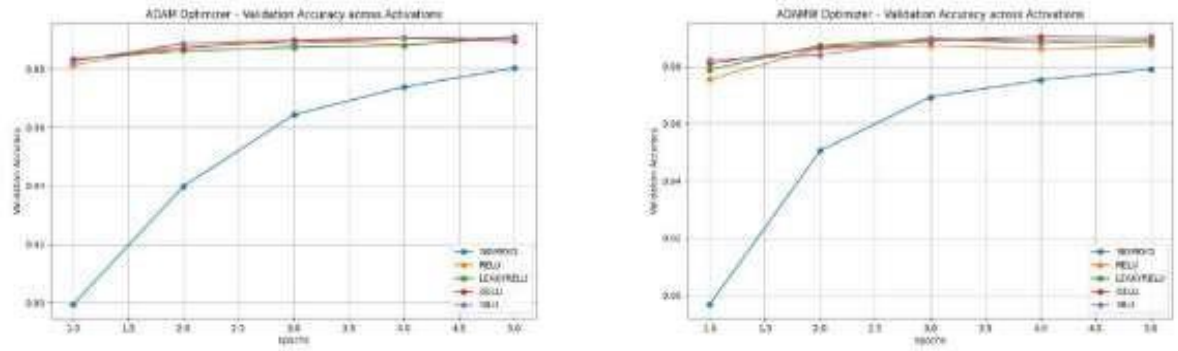


Figure 2: Activation Function Performance with Other Optimizers. Left: Performance under RMSProp, highlightingitshighvarianceandfailurewithSigmoid.Right:PerformanceunderSGDM,confirmingthe slow convergence rate across all AFs.

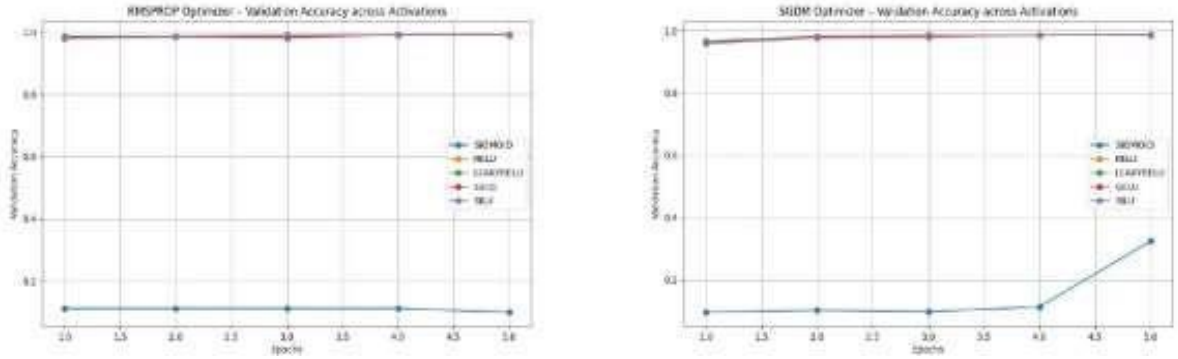


Figure 3: Optimizer Comparison: LeakyReLU (Left) and GeLU (Right). These plots demonstrate the mandatory speed advantage of adaptive methods (Adam, AdamW, RMSProp) over SGDM when paired with stable modern activations.

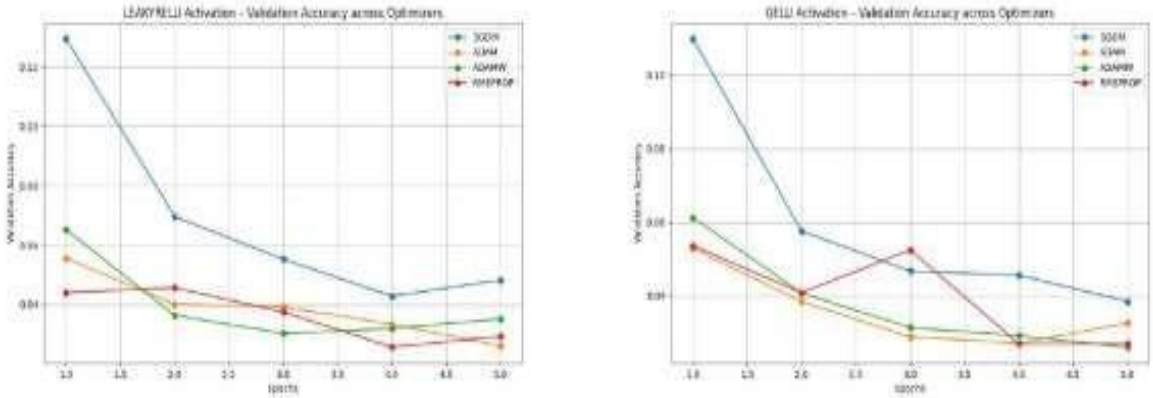


Figure 4: Individual AF Performance Diagnostics. Left: Performance of Sigmoid across all optimizers, clearlyshowingitslowaccuracyceiling.Right:PerformanceofSiLUacrossalloptimizers,demonstrating its strong and stable convergence behavior.

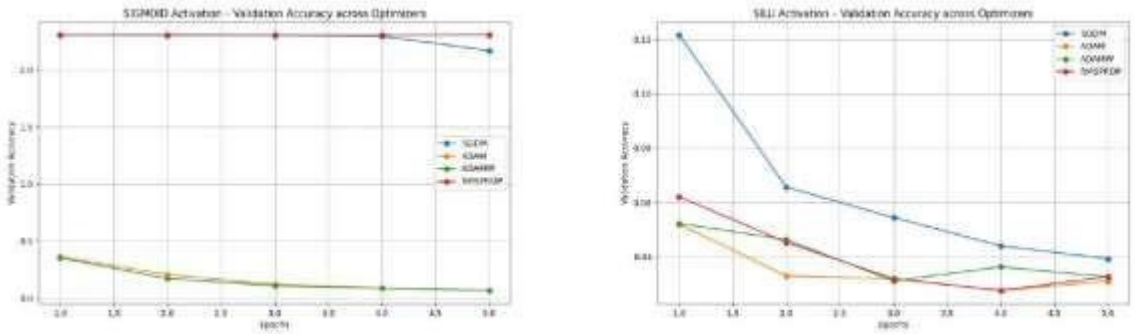
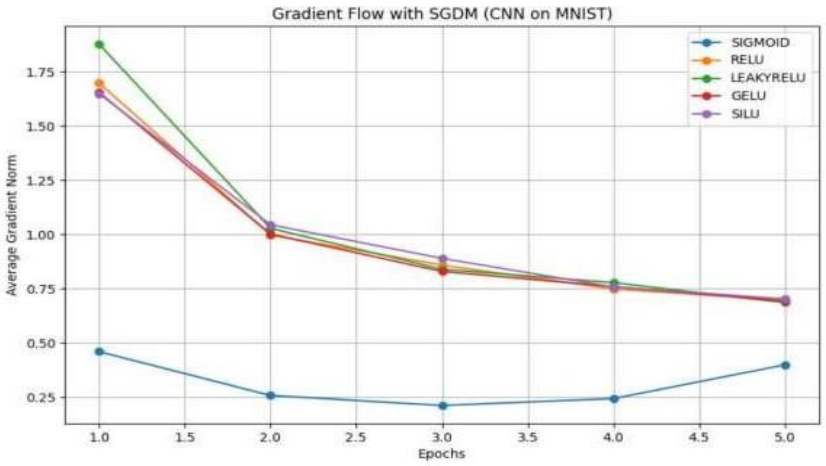


Figure 5: Gradient Flow Comparison under SGDM. Average L2 Norm of gradients for all AFs using the SGDM optimizer. The low magnitude and slow decay of the Sigmoid curve confirm the Vanishing Gradient Problem.



10. DISCUSSION

The key findings from the experiments strongly influence the selection criteria for building robust neural networks:

- Failure of Legacy Components: The Sigmoid function demonstrated severe gradient vanishing, yield- ing catastrophic accuracy with SGDM (0.3241) and complete failure with RMSProp (0.1009). The low gradient norms associated with Sigmoid were insufficient to sustain effective learning, highlighting its obsolescence.

- Best Performance and Stability: Modern activation functions-Leaky ReLU and GELU-achieved the highest validation accuracies (0.9910and 0.9918, respectively). Crucially, these functions areveri- fied to maintain 0 dead neurons, confirming their superiority over ReLU in stability.

- Dominance of Adaptive Optimizers: Adaptive optimizers (Adam, AdamW, RMSProp) consistently outperformed SGDM, confirming that dynamic learning rate adjustment is essential for rapid and stable convergence. RMS Prop displayed the highest variance, highlighting its sensitivity to activation- gradient interactions.

- Optimal Pair Identification: The configuration LeakyReLU + Adam achieved the best overall empirical validation accuracy (0.9910). However, the stability and generalization benefits of AdamW (decoupled weight decay) make the LeakyReLU/GELU + AdamW pairing the most highly recommended choice for applications prone to overfitting (e.g., CIFAR-10)

11.CONCLUSION

Based on the rigorous empirical analysis of activation function and optimizer synergy, this study provides definitive guidelines for enhancing the stability and efficiency of deep neural network training

11.1 Synthesis of Findings

The research successfully quantified the solutions to the core training challenges:

- Neuronal Stability: The data proved that LeakyReLU and GELU ensure perfect stability (0 dead neu ron count), eliminating the core ReLU instability observed in comparative runs

- Optimal Performance: The pairing of a stable activation function with the best adaptive optimizer yielded the highest performance ceiling, with Leaky ReLU + Adam achieving the highest empirical validation accuracy (0.9910on MNIST).

- Generalization: The AdamW optimizer consistently delivered superior convergence speed and generalization (via decoupled weight decay) compared to SGDM and standard Adam.

11.2 Final Recommendation

The definitive recommendation for maximizing performance, stability, and efficiency in Convolutional Neural Networks is the combination of:

- Activation Function: Leaky ReLU (or GELU)

- Optimizer: Adam W

This pairing creates a robust training environment that simultaneously avoids the critical failure modes of legacy components (vanishing gradients, dying neurons) while leveraging advanced adaptive scaling for accelerated learning and superior generalization

11. FUTURE SCOPE AND LIMITATIONS

The present study successfully analyzed the influence of various activation functions and optimizers on CNN training dynamics. However, there remains considerable scope for further exploration and refinement.

Extension to Deeper Architectures: Future research can extend this framework to more complex and deeper architectures such as ResNet, DenseNet, or Efficient Net, to evaluate the consistency of optimizer activation interactions in large-scale networks.

Exploration of Advanced Optimizers: New optimization algorithms such as Lion, AdaBelief, and Lookahead can be tested to compare their convergence be havior and generalization capabilities with traditional optimizers like Adam and RMSProp

Cross-Domain Experimentation: The experimental setup can be extended be yond image classification tasks to Natural Language Processing (NLP) and multi modal learning datasets, examining whether the observed trends hold across data modalities

Hardware and Computational Constraints: Due to limited computational resources, this study was restricted to small-scale datasets (MNIST and CIFAR-10). Future work could leverage GPU clusters or cloud platforms for large-scale training and hyper parameter tuning.

Theoretical Analysis: Further theoretical investigation of loss landscape geome try under different activation–optimizer combinations may provide deeper mathematical insights into convergence and stability behavior.

REFERENCES

1. *R.Kashyap,"Asurveyofdeeplearningoptimizers–firstandsecondordermethods,"arXivpreprint arXiv:2211.15596,2022.*
2. *C.Nwankpa,"Activationfunctions:Comparisonoftrendsinpracticeandresearchfordeeplearning,"arXivp reprintarXiv:1811.03378, 2018.*
3. *R.Abdulkadirov,P.Lyakhov,andN.Nagornov,"Surveyofoptimizationalgorithmsinmodernneural networks," Mathematics, vol. 11, no. 11, p. 2466, 2023.*
4. *S.R.Dubey,S.K.Singh,andB.B.Chaudhuri,"Activationfunctionsindeeplearning:A comprehensive survey and benchmark," Neurocomputing, vol. 503, pp. 92–108, 2022.*
5. *M. Lee, "Mathematical analysis and performance evaluation of the gelu activation function in deep learning," Journal of Mathematics, vol. 2023, no. 1, p. 4229924, 2023.*

6. *V.KuncandJ.Kl´ema, "Threedecadesofactivations:Acomprehensivesurveyof400activation functions for neural networks," arXiv preprint arXiv:2402.09092, 2024.*
7. *I.Loshchilov and F. Hutter, "Decoupled weight decay regularization," arXiv preprint arXiv:1711.05101,2017.*
8. *S. Selvakumari and M. Durairaj, "A comparative study of optimization techniques in deep learning using the mnist dataset," Indian Journal of Science and Technology, vol. 18, no. 10, pp. 803–810, 2025. [9]*
9. *M. Hammad, "Comprehensive survey of complex-valued neural networks: Insights into backpropagation and activation functions," arXiv preprint arXiv:2407.19258, 2024.*
10. *G. Bingham and R. Miikkulainen, "Efficient activation function optimization through surrogate modeling," Advances in Neural Information Processing Systems, vol. 36, pp. 6634–6661, 2023.*
11. *L.Emma,"Buildingandtrainingneuralnetworks:Architectures,optimization,andchallenges,"2022.*
12. *Y. Gong, Z. Lin, and J. Sun, "Tssr: A truncated and signed square root activation function for neural networks," in 2023 9th International Conference on Computer and Communications (ICCC), pp. 1978–1982, IEEE, 2023*