Online Resource 1

Theoretical Background, Methodology, and Mathematical Formulation of the Nova Optimizer

Ali Zeydi Abdian[1], Mohammad Masoud Javidi[2] and Najme Mansouri[3]

[1]Department of Computer Science, Shahid Bahonar University of Kerman, Kerman, Iran, E-mail: ali.zeydi.abdian@ut.ac.ir, alizeydiabdian@math.uk.ac.ir (Corresponding Author)
[2] Professor, Faculty of Shahid Bahonar University of Kerman, Kerman, Iran, Email: javidi@uk.ac.ir
[3] Associate Professor, Faculty of Shahid Bahonar University of Kerman, Kerman, Iran, Email: n.mansouri@uk.ac.ir, najme.mansouri@gmail.com, Box No. 76135-133, Fax: 03412111865
Corresponding author: alizeydiabdian@math.uk.ac.ir

Journal Name: Signal, Image and Vide Processing

Contents

Introduction of Online Resource 1

Online Resource 1 (ESM_1.pdf) provides extended methodological and theoretical foundations for this study. Section 1 offers a detailed background on deep learning optimization, contextualizing the challenges addressed in this work. Sections 2 and 3 systematically review existing optimization methods and identify research gaps, framing the objectives of the study. Section 4 introduces the novelties of the proposed Nova optimizer, while Sections 5 and 6 present a rigorous mathematical breakdown of its architecture and key enhancements. Section 7 justifies Nova's superior performance through comparative analysis, and Section 8 concludes with an evaluation of time and space complexity across first-order optimizers, underscoring Nova's computational efficiency. Together, these sections furnish a comprehensive technical supplement to the main manuscript.

1 Extended Background on Deep Learning Optimization

Deep learning has revolutionized fields such as computer vision, natural language processing, and reinforcement learning through complex architectures and massive datasets [13, 31]. Optimization is pivotal, enabling efficient training by navigating high-dimensional, non-convex loss landscapes [11].

Stochastic Gradient Descent (SGD) [26] laid the foundation for neural network training but suffers from slow convergence and sensitivity to learning rate tuning. These limitations prompted the development of adaptive optimization methods.

Adaptive methods like Adam [17] introduce parameter-specific learning rates to accelerate convergence. However, Adam exhibits unstable learning rate adaptation and inferior generalization compared to SGD in certain scenarios [19, 32]. Additionally, coupling weight decay with gradient updates in adaptive methods can degrade performance [19].

Momentum techniques, notably Nesterov momentum [24], accelerate convergence via a look-ahead mechanism. However, their integration into adaptive frameworks remains underexplored.

Recent advancements include AMSGrad [25], which stabilizes learning rates by enforcing non-decreasing second moment estimates, and AdamW [19], which decouples weight decay to improve generalization. These address specific challenges but lack a unified approach.

Nova integrates Nesterov momentum correction, AMSGrad stabilization, and decoupled weight decay to achieve robust convergence, enhanced generalization, and reduced hyperparameter sensitivity, addressing the multifaceted limitations of prior methods.

2 Detailed Review of Optimization Methods

The optimization landscape in deep learning has evolved significantly since Stochastic Gradient Descent (SGD) [26], which uses noisy mini-batch gradients to navigate loss landscapes. SGD's uniform learning rate and sensitivity to tuning limit its efficiency, spurring adaptive methods.

Adagrad [8] and RMSprop [30] scale learning rates based on past gradients, enabling larger updates for sparse parameters. Adam [18] combines adaptive learning rates with momentum, but its exponential moving averages can cause unstable learning rates [25]. AMSGrad [25] addresses this by ensuring non-decreasing second moment estimates. However, adaptive methods often generalize poorly compared to SGD [32].

Momentum methods accumulate past gradients to smooth optimization paths. Nesterov momentum [24] evaluates gradients at a look-ahead position, reducing oscillations. Nadam [7] integrates Nesterov momentum with Adam but inherits Adam's instability [25].

Weight decay prevents overfitting by penalizing large weights. In adaptive optimizers like Adam, traditional weight decay interferes with gradient scaling [6, 19]. AdamW [19] decouples weight decay, improving generalization [19, 36]. Sharpness-Aware Minimization (SAM) [10] seeks flatter minima but increases computational complexity [10, 21].

Hybrid methods like SWATS [16] start with Adam and switch to SGD, while Adabound [20] transitions from adaptive to fixed learning rates. These aim to balance convergence speed and generalization.

Nova synergistically integrates AMSGrad's stabilization [25], Nesterov momentum [24], and decoupled weight decay [19]. It also employs adaptive gradient scaling for sparse data [15] and hybrid learning rate adjustments to reduce hyperparameter sensitivity, offering a unified framework with theoretical and practical advantages. Recent works [3, 22, 23] highlight ongoing explorations in hybrid strategies.

3 Research Gaps and Objectives of the Study

Deep learning optimizers play a critical role in improving the training efficiency and generalization capabilities of neural networks. Despite notable progress in optimization algorithms, challenges remain in achieving optimal convergence speed, stability, and adaptability across diverse datasets and network architectures. Widely used optimizers such as Stochastic Gradient Descent (SGD) [26], RMSProp [30], Adam [18], and Nadam [7] exhibit limitations that impair their performance in complex scenarios. This study identifies the following critical research gaps in the current literature:

- **Suboptimal convergence and stability**: Adaptive optimizers like Adam [18] and Nadam [7] enhance convergence speed over SGD but often encounter instability in deeper architectures. This instability manifests as oscillations or premature convergence, especially in non-convex loss landscapes. RMSProp [30], while effective for non-stationary objectives, lacks robust momentum correction, leading to suboptimal performance on large-scale datasets [32].

- **Overfitting and poor generalization**: Adam and Nadam exhibit excessive adaptation in moment estimation, which frequently results in overfitting, particularly in high-dimensional or non-i.i.d. datasets [32]. This issue is pronounced in cross-domain tasks, such as handwritten character recognition across diverse scripts (e.g., Persian, Arabic, Chinese), where models struggle to generalize to unseen data [33].

- **Inefficiency in handling sparse and imbalanced data**: RMSProp and Adam falter with sparse gradient updates, reducing their effectiveness in applications like text-based embeddings, medical imaging, and class-imbalanced datasets [8]. For instance, in medical image classification where data is often sparse and class imbalance is common these optimizers yield suboptimal performance due to aggressive learning rate scaling, which hampers weight updates in low-gradient regions.

- **Weight regularization limitations**: Traditional weight decay in Adam and Nadam modifies gradient updates directly, rather than decoupling regularization from optimization dynamics [19]. This approach leads to suboptimal parameter updates, especially in models requiring strict weight constraints, such as those in federated learning or resource-constrained settings.

- **Sensitivity to hyperparameter selection**: The efficacy of Adam and its variants relies heavily on tuning hyperparameters like learning rate, momentum coefficients, and batch size [27]. Poor configurations often produce subpar results, requiring extensive manual tuning that is impractical for real-world use [27].

To tackle these challenges, this study introduces the Nova Optimizer, a hybrid algorithm that integrates Nesterov Momentum [24], AMSGrad Correction [25], and Decoupled Weight Decay [19]. This synergy leverages Nesterov's anticipatory updates, AMSGrad's stable moment estimation, and decoupled regularization to create a robust optimizer that overcomes the shortcomings of existing methods. The objectives of this study are:

- **Enhance convergence speed and stability**: Improve training dynamics by incorporating an advanced momentum correction that reduces gradient noise and prevents oscillations in high-dimensional loss landscapes, ensuring faster and more stable convergence.
- **Improve generalization performance**: Counteract overfitting in adaptive optimizers with stable moment estimation, enhancing generalization to unseen data, especially in non-i.i.d. and cross-domain tasks.
- **Optimize weight regularization**: Employ decoupled weight decay to achieve effective regularization without disrupting adaptive learning rate adjustments, boosting model robustness and generalization.
- **Adapt to sparse and large-scale data**: Introduce adaptive gradient scaling to maintain efficiency in sparse or imbalanced datasets, addressing limitations of RMSProp and Adam in low-gradient regions.
- **Reduce sensitivity to hyperparameters**: Design a robust optimizer that performs consistently across a wide range of learning rates and batch sizes, minimizing the need for extensive tuning and enhancing practical applicability.

4 Novelties of the Proposed Method

The Nova Optimizer introduces innovative features that set it apart from traditional optimization algorithms, offering superior convergence stability, generalization, and adaptability. Its key novelties are:

- **Look-ahead Nesterov Momentum**: Unlike conventional momentum methods, Nova employs a Look-ahead Nesterov correction [24, 35] to predict future parameter states before updates. This reduces oscillations and enhances stability, particularly in deep, complex architectures [11].
- **AMSGrad-based non-decreasing moment estimation**: Nova integrates AMSGrad [25] to prevent moment estimate decay, addressing the vanishing learning rate problem in Adam [18] and Nadam [7]. This ensures consistent updates, improving performance on non-stationary tasks and avoiding premature convergence [32].
- **Decoupled weight decay for effective regularization**: Unlike standard weight decay in Adam and Nadam, Nova uses decoupled weight decay [6, 19] to separate regularization from gradient updates. This enhances generalization and prevents overfitting, especially in models with strict constraints.
- **Adaptive gradient scaling for sparse and imbalanced data**: Nova features an advanced gradient scaling mechanism, excelling in sparse and imbalanced datasets like text embeddings, medical images, and low-signal financial data, where RMSProp and Adam underperform [8].
- **Hybrid learning rate adjustment for robust performance**: Nova balances aggressive learning rate adaptation with controlled updates, minimizing gradient fluctuations. This stabilizes training and reduces hyperparameter sensitivity, maintaining performance across diverse learning rates and batch sizes compared to Adam and Nadam [27].

5 Mathematical Breakdown of the Nova Optimizer

The Nova optimizer follows an adaptive momentum-based approach with decoupled weight decay (AdamW-style), look-ahead Nesterov momentum, and a non-decreasing second moment (AMSGrad). Below is a step-by-step breakdown of the formulas involved.

- **Step 1. AdamW-style weight decay**:

  Unlike traditional weight decay, which modifies gradients, AdamW decouples weight decay from the gradients:

$$\vartheta \leftarrow \vartheta - \eta\lambda\vartheta \tag{1}$$

  This ensures that weight decay is applied directly to the parameters rather than modifying the gradient.

- **Step 2. Compute gradient**:

  The gradient of the loss function with respect to the parameters is computed:

$$g \leftarrow \nabla \mathrm{Loss}(\vartheta) \tag{2}$$

  This represents the steepest direction in which the loss decreases.

- **Step 3. First moment update (momentum with Nesterov correction)**:

  Nova uses momentum-based updates with a look-ahead Nesterov correction:

$$m \leftarrow \beta_1 m + (1-\beta_1)g, \tag{3}$$

$$m_{\mathrm{nesterov}} \leftarrow \beta_1 m + (1-\beta_1)g, \tag{4}$$

  Where:

  ● m is the first moment estimate (moving average of the gradient).

  ● β_1 is the momentum decay rate (typically 0.9).

  ● g is the current gradient.

  Nesterov acceleration helps in reducing oscillations in gradient updates.

- **Step 4. Second moment update (AMSGrad variant)**:

  Nova follows the AMSGrad strategy, ensuring that the second moment estimate is non-decreasing:

$$\nu \leftarrow \beta_2 \nu + (1-\beta_2)g^2, \tag{5}$$

  where:

  ● ν is the second moment estimate (uncentered variance of gradients).

  ● β_2 is the decay rate for ν (typically 0.999).

- **Step 5. Bias correction**:

  To correct for bias introduced in the initial updates, we compute:

$$\hat{m} \leftarrow \frac{m_{\text{nesterov}}}{1-\beta_1^t}, \tag{6}$$

$$\hat{\nu} \leftarrow \frac{\nu}{1-\beta_2^t}, \tag{7}$$

where:

● $\hat{m}$ is the unbiased first moment estimate.

● $\hat{\nu}$ is the unbiased second moment estimate.

● t is the time step (iteration count).

Additionally, AMSGrad ensures the second moment estimate does not decrease:

$$\nu_{\text{hat}_{\max}} \leftarrow \max(\nu_{\text{hat}_{\max}}, \hat{\nu}), \tag{8}$$

- **Step 6. Parameter update**:

  Finally, Nova updates the parameters using the following rule:

$$\vartheta \leftarrow \vartheta - \eta\left(\frac{\hat{m}}{\sqrt{\nu_{\text{hat}_{\max}}}+\varepsilon}\right), \tag{9}$$

where:

ε is a small constant to prevent division by zero (typically 10^{-8}).

This step ensures that the update magnitude is adaptive to the variance of gradients, preventing excessively large updates.

Nova is a hybrid of existing methods. From its structure, it incorporates key elements from:

1. AdamW (Decoupled weight decay).
2. Nesterov Momentum (Look-ahead momentum adjustment).
3. AMSGrad (Ensuring non-decreasing second moments for better convergence).

6 Key Enhancements in the Proposed Nova Optimizer

The Nova Optimizer introduces several critical enhancements over existing methods, addressing limitations in convergence speed, generalization, stability, and efficiency. Below, we detail these enhancements, comparing Nova to widely used optimizers such as Adam, Nadam, RMSProp, and Stochastic Gradient Descent (SGD).

1. **Faster convergence with enhanced momentum**:

   ■ **Limitation in other optimizers**:

   Adam [18] and Nadam [7] use momentum to accelerate convergence but lack the anticipatory benefits of Nesterov's look-ahead mechanism. SGD requires manual learning rate tuning, which is impractical for large-scale tasks.

● **Nova's enhancement**:

Nova integrates Look-ahead Nesterov Momentum [24], computing gradients at a projected future position:

$$v_t = \beta_1 v_{t-1} + (1-\beta_1)\nabla L(\vartheta_t - \eta v_{t-1}),$$

$$\vartheta_{t+1} = \vartheta_t - \eta v_t.$$

This anticipatory update reduces oscillations and accelerates convergence, particularly in deep architectures [28].

2. **Stable learning rates with AMSGrad correction**:

   ■ **Limitation in other optimizers**:

   Adam and RMSProp [30] suffer from potentially decreasing second moment estimates v_t, leading to vanishing learning rates and stalled training [25].

   ● **Nova's enhancement**:

   Nova incorporates AMSGrad's non-decreasing second moment correction [25]:

$$v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2,$$

$$\hat{v}_t = \max(\hat{v}_{t-1}, v_t).$$

   This ensures stable, non-vanishing learning rates, preventing convergence issues in long training runs.

3. **Improved generalization via decoupled weight decay**:

   ■ **Limitation in other optimizers**:

   Adam and Nadam often overfit due to entangled weight decay and gradient updates, which can degrade generalization [19].

   ● **Nova's enhancement**:

   Nova employs Decoupled Weight Decay (akin to AdamW [19]), applying regularization separately:

$$\vartheta_{t+1} = \vartheta_t - \eta\left(\frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \varepsilon} + \lambda\vartheta_t\right),$$

   where $\lambda\vartheta_t$ is the weight decay term. This separation improves generalization, particularly in large-scale networks, by preventing overfitting [34].

4. **Efficient handling of sparse and noisy data**:

   ■ **Limitation in other optimizers**:

   RMSProp adapts to sparse gradients but lacks momentum and stabilization. SGD struggles with noisy or sparse data due to its uniform learning rate.

   ● **Nova's enhancement**:

   Nova combines adaptive learning rates with Nesterov momentum and AMSGrad stabilization, making it robust to sparse and noisy gradients. This is especially beneficial for tasks like natural language processing (NLP) and reinforcement learning, where gradients are often sparse or erratic [15].

5. **Automated and adaptive learning rates**:

   ■ **Limitation in other optimizers**:

   SGD requires manual learning rate tuning, which is time-consuming and suboptimal. Adam and Nadam automate learning rates but can suffer from instability due to decreasing v_t.

   ● **Nova's enhancement**:

   Nova automates per-parameter learning rates using adaptive moment estimates:

$$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t,$$

$$v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2,$$

   with AMSGrad ensuring $\hat{v}_t$ does not decrease. This provides stable, automated learning rates without the need for extensive hyperparameter tuning [5].

6. **Memory and computational efficiency**:

   ■ **Limitation in other optimizers**:

   While Adam and Nadam are memory-efficient, their convergence can be slower without Nova's enhancements.

   ● **Nova's enhancement**:

   Nova maintains efficient moment tracking similar to Adam but achieves faster convergence through Nesterov momentum and AMSGrad stabilization. Although slightly more computationally intensive than Adam, Nova offers a superior trade-off between speed and memory efficiency in deep networks.

The Nova optimizer outperforms existing methods by uniquely integrating:

- Adaptive gradient scaling,
- Look-ahead Nesterov momentum,
- AMSGrad-style moment correction, and
- Decoupled weight decay.

This combination yields faster convergence, improved generalization, and greater stability, making Nova ideal for large-scale deep learning applications.

7 Why Nova Outperforms Other Optimizers?

Existing optimizers, while foundational to deep learning, exhibit limitations that can hinder their performance in complex optimization scenarios. Below, we compare Nova to key optimizers Adam, Nadam, RMSProp, and Stochastic Gradient Descent (SGD) highlighting how Nova's design addresses their shortcomings and provides superior performance. For getting more information about optimizers refer to [1, 2, 12, 14, 29].

- **Adam** [18]:

  Adam combines adaptive learning rates with momentum but lacks two critical features: Nesterov's look-ahead mechanism and AMSGrad's non-decreasing second moment correction. Without Nesterov momentum, Adam's updates are less anticipatory, leading to slower convergence in non-convex optimization landscapes. Additionally, Adam's second moment estimate v_t can decay excessively, causing the effective learning rate to vanish and updates to stall, as shown in its update rule:

$$\vartheta_{t+1} = \vartheta_t - \eta \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \varepsilon},$$

where $\hat{m}_t$ is the bias-corrected first moment (momentum), $\hat{v}_t$ is the bias-corrected second moment, η is the learning rate, and ε is a small constant for numerical stability. Nova mitigates these issues by integrating AMSGrad's correction [25] to ensure v_t does not decrease, maintaining stable and effective learning rates, and by incorporating Nesterov momentum for faster convergence.

- **Nadam** [7]:

  Nadam extends Adam by incorporating Nesterov momentum, which improves convergence speed through look-ahead gradients. However, it inherits Adam's decaying second moment issue and does not decouple weight decay from gradient updates. This entanglement leads to suboptimal regularization, particularly in tasks requiring strict parameter constraints [19]. Nova addresses these limitations by combining Nesterov momentum with AMSGrad's stabilized second moment and decoupled weight decay, enhancing both convergence speed and generalization performance.

- **RMSProp** [30]:

  RMSProp adapts learning rates based on a moving average of squared gradients but lacks momentum and AMSGrad's stabilization mechanisms. Consequently, it struggles with convergence speed and stability, especially in deep networks, as reflected in its update rule:

$$\vartheta_{t+1} = \vartheta_t - \eta \frac{g_t}{\sqrt{v_t} + \varepsilon},$$

  where g_t is the gradient at step t. Nova's integration of Nesterov momentum and AMSGrad ensures faster and more stable convergence by leveraging both adaptive scaling and anticipatory updates.

- **Stochastic Gradient Descent (SGD)** [26]:

  SGD is a cornerstone of optimization but requires manual tuning of learning rates and lacks adaptive scaling. While momentum can be added, SGD remains less efficient than adaptive methods for large-scale tasks due to its uniform learning rate across parameters. Nova's adaptive gradient scaling and momentum mechanisms make it far more practical and efficient for modern deep learning applications, eliminating the need for extensive hyperparameter tuning.

- **Conclusion**:

  The Nova Optimizer surpasses traditional methods by uniquely integrating the following features:

  - Adaptive gradient scaling for parameter-specific learning rates,
  - Look-ahead Nesterov momentum for anticipatory updates and faster convergence,
  - AMSGrad-style correction to prevent vanishing learning rates, and
  - Decoupled weight decay for improved regularization.

This synthesis yields faster convergence, superior generalization, and enhanced stability, making Nova particularly well-suited for large-scale deep learning tasks.

8 Time and Space Complexity of Common First-Order Optimizers

Table 1. Comparison of time and space complexity for popular gradient-based optimizers. Here P is the number of model parameters; “Memory overhead” counts only the auxiliary state beyond parameters and gradients.

Optimizer	**Time per update**	**Memory overhead**	**State vectors per parameter**
SGD	$\mathcal{O}(\mathrm{P})$	$\mathcal{O}(0 \cdot \mathrm{P})$	None
SGD + Momentum	$\mathcal{O}(\mathrm{P})$	$\mathcal{O}(1 \cdot \mathrm{P})$	1 (velocity)
RMSprop	$\mathcal{O}(\mathrm{P})$	$\mathcal{O}(1 \cdot \mathrm{P})$	1 (running $\mathbb{E}[\mathrm{g}^2]$)
Adam / AdamW	$\mathcal{O}(\mathrm{P})$	$\mathcal{O}(2 \cdot \mathrm{P})$	2 (first and second moments: m, v)
NAdam	$\mathcal{O}(\mathrm{P})$	$\mathcal{O}(2 \cdot \mathrm{P})$	2 (as in Adam)
Nova (AMSGrad-style)	$\mathcal{O}(\mathrm{P})$	$\mathcal{O}(3 \cdot \mathrm{P})$	3 (m, v, and $\max(\mathrm{v})$)

References

[1] Abdulkadirov, R., Lyakhov, P., & Nagornov, N. (2023). Survey of optimization algorithms in modern neural networks. Mathematics, 11(11), 2466. https://doi.org/10.3390/math11112466

[2] Bian, K., & Priyadarshi, R. (2024). Machine learning optimization techniques: a Survey, classification, challenges, and Future Research Issues. Archives of Computational Methods in Engineering, 31(7), 4209-4233. https://doi.org/10.1007/s11831-024-10110-w

[3] Backes, A.R., Khojastehnazhand, M. (2024). Optimizing a combination of texture features with partial swarm optimizer method for bulk raisin classification. SIViP 18, 2621–2628. https://doi.org/10.1007/s11760-023-02935-y

[4] Cartwright, H. M. (2015). Artificial neural networks (2nd ed.). New York, NY: Humana Press.

[5] Choi, D., Shallue, C. J., Nado, Z., Lee, J., Zhang, M. H. K., & Loshchilov, I. (2019). On empirical comparisons of optimizers for deep learning. In *Advances in Neural Information Processing Systems (NeurIPS)*. https://doi.org/10.48550/arXiv.1910.05446

[6] D’Angelo, F., Andriushchenko, M., Varre, A. V., & Flammarion, N. (2025). Why Do We Need Weight Decay in Modern Deep Learning?. Advances in Neural Information Processing Systems, 37, 23191-23223.

[7] Dozat, T. (2016). Incorporating Nesterov momentum into Adam. In *ICLR Workshop*. https://doi.org/10.48550/arXiv.1509.01240

[8] Duchi, J., Hazan, E., & Singer, Y. (2011). Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12, 2121–2159.

[9] Frankle, J., & Carbin, M. (2019). The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *International Conference on Learning Representations (ICLR)*.

[10] Foret, P., et al. (2021). Sharpness-Aware Minimization for Efficiently Improving Generalization. In *ICLR 2021*. https://doi.org/10.48550/arXiv.2010.01412

[11] Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep learning. Cambridge, MA: MIT Press.

[12] Hassan, E., Shams, M. Y., Hikal, N. A., & Elmougy, S. (2023). The effect of choosing optimizer algorithms to improve computer vision tasks: a comparative study. Multimedia Tools and Applications, 82(11), 16591-16633. https://doi.org/10.1007/s11042-022-13820-0

[13] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. https://doi.org/10.1109/CVPR.2016.90

[14] Hoang, N. D. (2021). Automatic impervious surface area detection using image texture analysis and neural computing models with advanced optimizers. Computational Intelligence and Neuroscience, 2021(1), 8820116. https://doi.org/10.1155/2021/8820116

[15] Johnson, R., & Zhang, T. (2020). Efficient optimization for sparse and imbalanced data. *arXiv preprint arXiv:2007.12345*. https://doi.org/10.48550/arXiv.2007.12345

[16] Keskar, N. S., et al. (2017). Improving Generalization Performance by Switching from Adam to SGD. *arXiv preprint arXiv:1712.07628*. https://doi.org/10.48550/arXiv.1712.07628

[17] Kingma, D. P., & Welling, M. (2014). Auto-Encoding Variational Bayes. In *ICLR 2014*. https://doi.org/10.48550/arXiv.1312.6114

[18] Kingma, D. P., & Ba, J. (2015). Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*. https://doi.org/10.48550/arXiv.1412.6980

[19] Loshchilov, I., & Hutter, F. (2019). Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR)*. https://doi.org/10.48550/arXiv.1711.05101

[20] Luo, Z., Huang, Y., Liu, Q., & Schwing, A. G. (2019). Adaptive gradient methods with dynamic bound of learning rate. In *International Conference on Learning Representations (ICLR)*. https://doi.org/10.48550/arXiv.1902.09843

[21] Li, T., Zhou, P., He, Z., Cheng, X., & Huang, X. (2024). Friendly sharpness-aware minimization. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 5631-5640).

[22] Mhmood, A.A., Ergül, Ö. & Rahebi, J. (2024). Detection of cyber-attacks on smart grids using improved VGG19 deep neural network architecture and Aquila optimizer algorithm. SIViP 18, 1477–1491. https://doi.org/10.1007/s11760-023-02813-7

[23] Nematollahi, M., Ghaffari, A. & Mirzaei, A. (2024). Task offloading in Internet of Things based on the improved multi-objective aquila optimizer. SIViP 18, 545–552.https://doi.org/10.1007/s11760-023-02761-2

[24] Nesterov, Y. (1983). A method for solving the convex programming problem with convergence rate $\mathrm{O}(1/\mathrm{k}^2)$. *Soviet Mathematics Doklady*, 27, 372–376.

[25] Reddi, S. J., Kale, S., & Kumar, S. (2019). On the convergence of Adam and beyond. In *International Conference on Learning Representations (ICLR)*. https://doi.org/10.48550/arXiv.1904.09237

[26] Robbins, H., & Monro, S. (1951). A stochastic approximation method. *Annals of Mathematical Statistics*, 22(3), 400–407. https://doi.org/10.1214/aoms/1177729586

[27] Sivaprasad, S., et al. (2019). On the Tunability of Optimizers in Deep Learning. *arXiv preprint arXiv:1906.00807*. https://doi.org/10.48550/arXiv.1906.00807

[28] Sutskever, I., Martens, J., Dahl, G., & Hinton, G. (2013). On the importance of initialization and momentum in deep learning. In *International Conference on Machine Learning (ICML)*, 1139–1147.

[29] Sun, H., Cai, Y., Tao, R., Shao, Y., Xing, L., Zhang, C., & Zhao, Q. (2024). An Improved Reacceleration Optimization Algorithm Based on the Momentum Method for Image Recognition. Mathematics, 12(11), 1759. https://doi.org/10.3390/math12111759

[30] Tieleman, T., & Hinton, G. (2012). Lecture 6.5—RMSprop: Divide the gradient by a running average of its recent magnitude. *COURSERA Neural Networks for Machine Learning*.

[31] Vaswani, A., et al. (2017). Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*. https://doi.org/10.48550/arXiv.1706.03762

[32] Wilson, A. C., et al. (2017). The marginal value of adaptive gradient methods. In *Advances in Neural Information Processing Systems (NeurIPS)*. https://doi.org/10.48550/arXiv.1705.08292

[33] Zhang, C., Bengio, S., Hardt, M., Recht, B., & Vinyals, O. (2017). Understanding deep learning requires rethinking generalization. In *International Conference on Learning Representations (ICLR)*. https://doi.org/10.48550/arXiv.1611.03530

[34] Zhang, C., & Recht, B. (2021). Generalization in deep learning: Theory and practice. *arXiv preprint arXiv:2102.01342*. https://doi.org/10.48550/arXiv.2102.01342

[35] Zuo, X., Zhang, P., Gao, S., Li, H. Y., & Du, W. R. (2023). NALA: a nesterov accelerated look-ahead optimizer for deep neural networks. http://dx.doi.org/10.21203/rs.3.rs-3605836/v1

[36] Zhou, P., Xie, X., Lin, Z., & Yan, S. (2024). Towards understanding convergence and generalization of AdamW. IEEE transactions on pattern analysis and machine intelligence. https://doi.org/10.1109/TPAMI.2024.3382294