Nova: A Novel Optimizer Integrating Nesterov Momentum, AMSGrad, and Decoupled Weight Decay for Deep Learning

Ali Zeydi Abdian*, Mohammad Masoud Javidi†, Najme Mansouri‡

Abstract

The choice of optimization algorithm significantly impacts deep learning model performance, affecting convergence speed, generalization, and training stability. Existing optimizers, such as Adam, AMSGrad, and AdamW, face significant limitations that hinder their effectiveness across diverse deep learning tasks and architectures. These include suboptimal momentum utilization, stalled training due to vanishing learning rates, and compromised generalization with weight decay. To address these critical challenges, we propose Nova, a novel hybrid optimizer that synergistically integrates Nesterov momentum, AMSGrad's non-decreasing second moment estimate, and decoupled weight decay. Nova introduces a pioneering integration of established techniques, distinguishing it from prior optimizers. Beyond its core components, Nova incorporates adaptive gradient scaling to handle sparse and imbalanced data efficiently and a hybrid learning rate adjustment to reduce hyperparameter sensitivity. This combination enhances training dynamics, stability, and robustness across various deep learning tasks. Experimental results on benchmark datasets, such as CIFAR-10, MNIST, SST-2, and noisy MNIST, demonstrate Nova's superior performance. For instance, Nova achieves a test accuracy of 90.01% on CIFAR-10, significantly outperforming Adam's 83.14% and Nadam's 80.85%. On the SST-2 dataset, Nova achieves a test accuracy of 82.00%, surpassing Adam's 66.28% and AdamW's 81.65%. Furthermore, Nova demonstrates robustness to noisy data, achieving a test accuracy of 96.58% on noisy MNIST. These results show that Nova effectively overcomes the challenges of slow convergence, poor generalization, and sensitivity to data characteristics faced by existing optimizers. Nova's exceptional performance across diverse benchmark datasets suggests its significant potential for broader applications in deep learning. By addressing the critical limitations of existing optimizers through a novel and synergistic combination of techniques, Nova represents a significant advancement in deep learning optimization, offering a powerful and robust solution that enhances training efficiency, model generalization, and stability. The code and additional experimental results related to this paper are publicly available at https://github.com/Aliyar4061/Nova-Optimizer/tree/main.

Keywords: Deep learning, Optimization, Adaptive Gradient Descent, AMSGrad, Nesterov Momentum, Weight Decay.

*Ali Zeydi Abdian, 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)

†Mohammad Masoud Javidi, Professor, Faculty of Shahid Bahonar University of Kerman, Kerman, Iran, Email: javidi@uk.ac.ir

‡Najme Mansouri, 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

1 Introduction

The rapid advancement of deep learning has revolutionized a wide array of fields, including computer vision, natural language processing, and reinforcement learning, by leveraging increasingly complex architectures and massive datasets [13, 31]. At the heart of this progress lies the optimization process, which is pivotal for training neural networks by efficiently navigating high-dimensional, non-convex loss landscapes [11]. Early approaches based on Stochastic Gradient Descent (SGD) [26] provided a solid foundation, yet their inherent limitations in convergence speed and sensitivity to learning rate tuning have spurred the development of adaptive optimization methods such as Adam [17].

Despite their widespread adoption, adaptive optimizers often encounter significant challenges. For example, while Adam introduces parameter-specific learning rates to accelerate convergence, it has been shown to exhibit unstable learning rate adaptation and inferior generalization performance compared to SGD in certain scenarios [19, 32]. Moreover, the coupling between weight decay and gradient updates in adaptive methods can lead to unintended interference, degrading the performance of the trained models [19]. Concurrently, momentum-based techniques, most notably Nesterov momentum [24], have demonstrated their potential to accelerate convergence through a look-ahead mechanism, yet their integration into adaptive frameworks remains underexplored.

Recent advancements have sought to mitigate these issues through hybrid strategies. AMSGrad [25] addresses the instability of adaptive learning rates by enforcing non-decreasing second moment estimates, while AdamW decouples weight decay from gradient updates to improve generalization. However, each of these improvements tackles only one aspect of the overall optimization challenge. As a result, there remains a need for a unified approach that simultaneously overcomes momentum inefficiency, learning rate instability, and regularization conflicts.

In this work, we propose Nova, a novel optimizer that cohesively integrates three critical innovations:

- **Nesterov momentum correction**: By incorporating a look-ahead mechanism, Nova enhances gradient estimation accuracy, thereby reducing oscillations and accelerating convergence.
- **AMSGrad stabilization**: Nova maintains non-decreasing second moment estimates, which prevents learning rate collapse and ensures stable adaptation during training.
- **Decoupled weight decay**: By separating weight decay regularization from the gradient update process, Nova mitigates the adverse interactions that can degrade model performance.

By synthesizing these components, Nova aims to achieve robust convergence, enhanced generalization, and reduced sensitivity to hyperparameter tuning. These improvements are essential for unlocking the full potential of deep learning models and for facilitating their deployment in real-world applications where training stability and performance are paramount.

Section 2 reviews related works, highlighting research gaps and the objectives of this study. Section 3 introduces the proposed Nova optimizer, detailing its methodology and key innovations. Section 4 describes the experimental setup, including datasets, evaluation metrics, and benchmark comparisons. Section 5 presents the results, covering performance metrics, learning dynamics, and classification analyses. Section 6 provides a comprehensive discussion of the findings, examining Nova's strengths and robustness. Finally, Section 7 concludes the study with key takeaways and future research directions.

2 Related Works

The landscape of deep learning optimization has evolved significantly since the early days of Stochastic Gradient Descent (SGD) [26]. While SGD introduced the fundamental concept of using noisy, mini-batch gradients to navigate the loss landscape and accelerated convergence compared to batch gradient descent, it is inherently limited by its reliance on a uniform learning rate across all parameters and its sensitivity to the choice of this rate, often requiring extensive manual tuning. This limitation spurred the development of adaptive gradient methods. Early adaptive approaches like Adagrad [8] and RMSprop [30] addressed the uniform learning rate problem by scaling learning rates for each parameter based on the history of squared gradients, allowing for larger updates in parameters with sparse or small gradients. Building on these, Adam [18] combined adaptive learning rates with momentum, quickly becoming a widely used optimizer due to its efficiency and generally good performance. However, Adam's reliance on exponential moving averages for the second moment estimate can, in certain scenarios, lead to unstable learning rates that may decrease prematurely, potentially stalling training [25]. This instability motivated the development of AMSGrad [25], which mitigates this issue by ensuring that the second moment estimates are non-decreasing. Despite the efficiency of adaptive methods, they have sometimes been observed to exhibit poorer generalization performance compared to traditional SGD, particularly in tasks where the interaction between momentum and gradient noise negatively impacts convergence to wide, flatter minima [32].

Alongside adaptive methods, omentum techniques have played a crucial role in accelerating convergence. Momentum methods accumulate past gradients to maintain direction and speed, dampening oscillations and smoothing the optimization path. Nesterov momentum [24], a notable variant, improves upon standard momentum by evaluating the gradient at a "look-ahead" position, leading to more accurate updates and further reducing oscillations. While effective, integrating the benefits of Nesterov momentum into adaptive frameworks like Adam has remained challenging. Nadam [7] was an attempt to combine Nesterov momentum with Adam, but it unfortunately retained Adam's vulnerability to the exponentially decaying second moment issue, potentially leading to similar instability problems as Adam [25].

Regularization and weight decay are also critical components in training deep learning models, primarily used to prevent overfitting and improve generalization. Weight decay works by adding a penalty to the loss function proportional to the magnitude of the weights. However, the direct integration of traditional weight decay with adaptive optimizers like Adam can lead to suboptimal performance. This is because the standard weight decay formulation becomes entangled with the adaptive scaling of gradients, effectively resulting in a non-uniform decay rate across parameters that can degrade the regularization effect [6, 19]. AdamW [19] addressed this by decoupling weight decay from the gradient updates, applying it directly to the parameters in a manner similar to SGD. This decoupled approach has been shown to improve generalization in many cases [19,36]. Other regularization strategies like Sharpness-Aware Minimization (SAM) [10] aim to improve generalization by seeking optimization paths that lead to flatter minima, which are believed to correspond to better generalization, but this often comes at the cost of increased computational complexity [10,21].

The ongoing challenges with convergence speed, stability, and generalization in deep learning optimization have led to the exploration of *hybrid and stabilized approaches*. Methods like SWATS [16] attempt to leverage the strengths of different optimizers by starting with Adam for fast initial convergence and switching to SGD later in training to potentially improve generalization. Adabound [20] is another hybrid approach that dynamically adjusts learning rates by transitioning from adaptive scaling in the early stages of training to fixed learning rates similar to SGD in later stages, aiming to achieve both fast convergence and good final performance. As previously mentioned, AMSGrad [25] specifically focuses on stabilizing the adaptive learning rate by ensuring the non-decreasing nature of the second moment estimate, addressing a key instability issue in Adam.

Our proposed Nova optimizer builds upon and synergistically integrates key advancements from these different lines of research to address the identified limitations in a unified framework. Nova leverages

AMSGrad's stabilized second moment estimates [25] to overcome Adam's instability and ensure robust convergence by preventing premature learning rate decay. It further enhances convergence speed and stability by incorporating Look-ahead Nesterov Momentum [24], which provides more accurate and oscillation-reducing updates than standard momentum or Adam's variant. Crucially, Nova integrates decoupled weight decay [19] to ensure effective regularization and improved generalization without the detrimental interference observed when traditional weight decay is used with adaptive gradient updates. By unifying the benefits of AMSGrad's stabilized learning rates, Nesterov momentum, and decoupled weight decay, along with adaptive gradient scaling for efficient handling of sparse and imbalanced data [15], and a hybrid learning rate adjustment to reduce hyperparameter sensitivity, Nova offers a cohesive framework that aims to provide both theoretical guarantees and significant practical advantages, directly addressing the multifaceted limitations found in previous approaches.

In conclusion, the development of optimization algorithms has significantly evolved from the early days of SGD to the advanced adaptive and hybrid methods used today. Nova, by building on the strengths of AMSGrad, Nesterov momentum, and decoupled weight decay in a novel synergistic manner, offers a promising direction for future research and practical applications. The ongoing exploration of new techniques and hybrid strategies, as highlighted by recent work such as [3, 22, 23], will continue to shape the field and drive advancements in machine learning optimization.

2.1 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.

2.2 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].

3 The Proposed Method

Algorithm 1 presents pseudocode related to Nova optimizer.

3.1 Pseudocode

Algorithm 1 Nova Optimizer

Require: Parameters ϑ, Learning rate η, Momentum decay rates β_1, β_2, Weight decay λ, Epsilon ε, Max iterations T

Ensure: Optimized parameters ϑ

Initialize:
  First moment vector: $m \leftarrow 0$
  Second moment vector: $v \leftarrow 0$
  Max second moment: $\hat{v}_{\text{hat\_max}} \leftarrow 0$
  Time step: $t \leftarrow 0$
for $t = 1$ to T **do**
  1. AdamW-style weight decay:
    $\vartheta \leftarrow \vartheta - \eta\lambda\vartheta$
  2. Compute gradient:
    $g \leftarrow \nabla \text{Loss}(\vartheta)$
  3. Update first moment (Nesterov adjustment):
    $m \leftarrow \beta_1 m + (1-\beta_1) g$
    $m_{\text{nesterov}} \leftarrow \beta_1 m + (1-\beta_1) g$
  4. Update second moment (AMSGrad):
    $v \leftarrow \beta_2 v + (1-\beta_2) g^2$
  5. Bias correction:
    $\hat{m} \leftarrow \dfrac{m_{\text{nesterov}}}{1-\beta_1^t}$
    $\hat{v} \leftarrow \dfrac{v}{1-\beta_2^t}$
    $v_{\text{hat\_max}} \leftarrow \max(v_{\text{hat\_max}}, \hat{v})$
  6. Parameter update:
    $\vartheta \leftarrow \vartheta - \eta \left(\dfrac{\hat{m}}{\sqrt{\hat{v}_{\text{hat\_max}}} + \varepsilon} \right)$
end for
return ϑ

3.2 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:

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

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

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).

3.3 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.

3.4 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.

3.5 Time and Space Complexity of Common First-Order Optimizers

Let P denote the total number of trainable parameters in a model. Each optimizer updates all parameters in one pass, performing a constant number of element-wise operations per parameter, so the *time complexity* per iteration is $\mathcal{O}(P)$.

Many optimizers maintain auxiliary state vectors of length P. If an optimizer keeps k such vectors, its *additional memory complexity* is:

$$\mathcal{O}(kP) \quad \text{(beyond the } P \text{ parameters and their gradients).}$$

Table 1 summarizes the relative computational efficiency and memory usage of various 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 $\max(\mathrm{v})$)

4 Experimental Setup

4.1 Datasets

In this subsection, the datasets that will be used to evaluate the model's performance are introduced. The images of each dataset are arranged according to their classes (labels).

4.1.1 CIFAR-10 Dataset

The CIFAR-10 (Canadian Institute For Advanced Research) dataset is a widely used benchmark in deep learning and computer vision, consisting of 60,000 32×32 color images across 10 classes, including airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck, with 6,000 images per class. It is divided into 50,000 training images and 10,000 test images, making it ideal for tasks like image classification and object recognition. Despite its low resolution, which can challenge fine detail recognition, the dataset is balanced and exhibits significant intra-class variability. Common preprocessing steps include normalization and data augmentation techniques like flipping and cropping. CIFAR-10 is often used to evaluate models such as CNNs, ResNet, and VGG, with accuracy being the primary performance metric. See Fig. 1a.

4.1.2 MNIST Dataset

The MNIST (Modified National Institute of Standards and Technology) database is defined as a dataset of handwritten digit images, composed of 60,000 training images and 10,000 test images. In the experiments, 40,000 samples from the training set were utilized, with digits 4 and 8 excluded, while 80 samples containing digits 4 and 8 were used for testing. The validation set was randomly selected and extracted from the original training set. A variety of English handwritten digits from the MNIST dataset, which includes 10 distinct classes, are displayed in Fig. 1b. Fig. 2 shows noisy MNIST after applaying GaussianNoise algorithm with noise_std = 0.5.

4.1.3 SST-2 Dataset

The Stanford Sentiment Treebank (SST-2) is a benchmark dataset widely employed in the evaluation of binary sentiment classification systems. Originally introduced by Socher et al. (2013) in the context of recursive deep models, SST-2 is a curated subset of the full Stanford Sentiment Treebank, distilled to address a binary classification task categorizing sentences as either positive or negative. The dataset comprises 6,920 training examples, 872 development samples, and 1,821 test samples, providing a balanced yet challenging corpus that contains a rich variety of linguistic constructs and sentiment nuances derived from movie reviews.

(a) CIFAR-10 (b) MNIST

Fig 1. Visualization of CIFAR-10 and MNIST datasets.

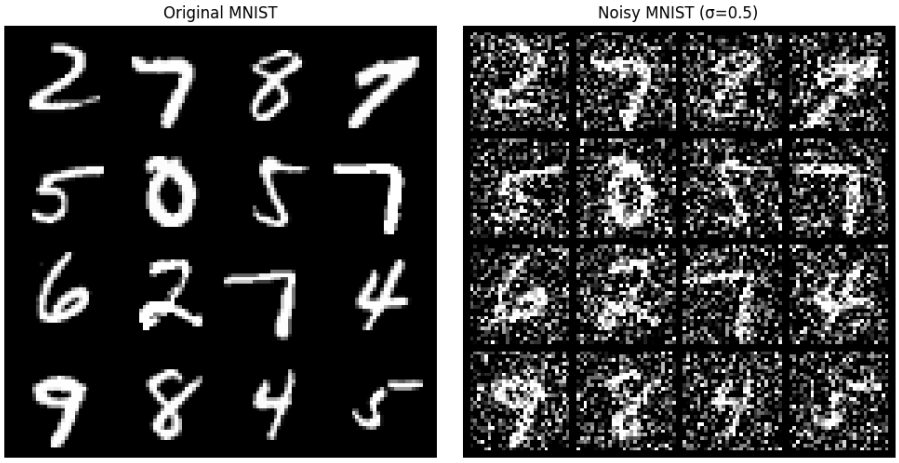


Fig 2. MNIST and noisy MNIST

4.1.4 Evaluating the Performance of Various Optimizers on Some Well-Known Networks: A Comparison with Nova

In this subsection, well-known neural networks will be introduced for evaluating the performance of different optimizers with Nova. ResNet-18 and LeNet-5 (refer to website Keras or Kaggle for seeing architecture) will be utilized. Tables 2, 3, 4, 5, 6, 7, 8 and 9 are provided to present details about their architectures, a comparison between them and configurations of the models. The code and additional experimental results related to this paper are publicly available at https://github.com/Aliyar4061/Nova-Optimizer/tree/main.

Table 2. ResNet-18 architecture details

Layer / Block	**Operation**	**Output / Notes**
Input	RGB Image	224×224×3
Conv1	7×7 Convolution, 64 filters, stride 2, padding 3	112×112×64
BN + ReLU	–	–
MaxPool	3×3 Max Pooling, stride 2	56×56×64
Residual Blocks (each block: 2 conv layers, BN, ReLU)		
Layer 1 (conv2_x)	2 blocks with 64 filters (stride 1)	56×56×64
Layer 2 (conv3_x)	2 blocks with 128 filters; first block uses stride 2	28×28×128
Layer 3 (conv4_x)	2 blocks with 256 filters; first block uses stride 2	14×14×256
Layer 4 (conv5_x)	2 blocks with 512 filters; first block uses stride 2	7×7×512
Global Average Pooling	Pooling over 7×7 spatial dimensions	1×1×512
FC Layer	Fully Connected, e.g. 1000 classes for ImageNet	1000 (or as specified)

Table 3. LeNet-5 architecture details

Layer	**Operation**	**Output / Notes**
Input	Grayscale image	32×32×1
C1	Convolution (5×5 kernel), 6 filters, activation tanh	28×28×6
S2	Average pooling (subsampling)	14×14×6
C3	Convolution (5×5 kernel), 16 filters, tanh	10×10×16
S4	Average pooling (subsampling)	5×5×16
C5	Convolution (fully-connected), kernel size matching input, tanh	1×1×120
F6	Fully connected	84 units
Output	Fully connected	10 units (for digit classification)

Table 4. Comparison of ResNet-18 and LeNet-5

Attribute	**ResNet-18**	**LeNet-5**
Year developed	2015	1998
Depth (Layers)	18	7
Key Innovation	Residual (skip) connections	Convolution + pooling
Original application	Large-scale image recognition	Handwritten digit recognition
Training techniques	Batch normalization, deeper architecture	Simpler activations (e.g., tanh)
Network complexity	High (designed for complex datasets)	Low (optimized for limited variability)

Table 5. Used architeture CIFAR-10 ResNet-18 in the paper

Layer/Block	**Modification**	**Parameters/Notes**
Input	CIFAR-10 Image	32×32×3
Conv1	3×3 Conv, stride 1	Original: 7×7 stride 2
MaxPool	Removed	Identity operation
Residual Blocks	Added Spatial Dropout	Layer1: 0.05, Layer2: 0.1 Layer3: 0.15, Layer4: 0.2
Classifier	Modified FC Layers Added Dropout	512 → 256 → 10 FC dropout: 0.3

Table 6. Used architeture MNIST LeNet-5 in the paper

Layer	Operation	Parameters
Input	MNIST Image	28×28×1
Conv1	5×5 Conv, 6 filters	Output: 24×24×6
MaxPool	2×2	Output: 12×12×6
Conv2	5×5 Conv, 16 filters	Output: 8×8×16
MaxPool	2×2	Output: 4×4×16
FC1	Linear Layer	256 → 120
FC2	Linear Layer	120 → 84
FC3	Linear Layer	84 → 10
Regularization	Dropout	p=0.25 after FC1

Table 7. Training configuration

Category	Parameter	Value
Training	Epochs	30
	Batch Size	128
	LR Scheduler	StepLR (gamma=0.9)
Optimizers	Base LR	0.001
	Weight Decay	0.01
	Nova	betas=(0.9, 0.999), eps=1e-8
	SGD	momentum=0.9
Data Augmentation	CIFAR-10	RandomCrop(32,4)+Flip
	MNIST	None
Normalization	CIFAR-10	Mean=[0.4914, 0.4822, 0.4465] Std=[0.2023, 0.1994, 0.2010]
	MNIST	Mean=0.1307, Std=0.3081

Table 8. Training configuration for SST-2

Category	Parameter	Value
Training	Epochs	10
	Batch Size	64
	LR Scheduler	StepLR (gamma=0.9)
Optimizers	Base LR	0.001
	Weight Decay	0.0
	Nova	betas=(0.9, 0.999), eps=1e-8
	Adam	Default parameters
Data Augmentation	SST-2	None
Normalization	Preprocessing	Lowercase conversion, tokenization, and padding
	Vocabulary Construction	Count-based, with ‘<unk>‘ and ‘<pad>‘ tokens

Table 9. Detailed architecture of SST-2 text classification model

Component	Description
Model Type	Feedforward Neural Network (Embedding + Linear Classifier)
Embedding Layer	• Input dimension: vocab_size (determined from training data) • Output dimension (embed_dim): 100 (configurable) • Special tokens: <unk>=0, <pad>=1 (for OOV words and padding) • Trainable parameters: vocab_size × embed_dim
Pooling	• Operation: Mean pooling across sequence length (dim=1) • Reduces variable-length sequences to fixed-size embedding
Classifier Layer	• Single Linear layer (embed_dim → num_class) • Output size: 2 (binary classification: Negative/Positive) • No explicit activation (CrossEntropyLoss includes LogSoftmax) • Trainable parameters: embed_dim × num_class + bias
Forward Pass	1. Tokenize text → convert to vocab indices 2. Embedding layer: maps indices to dense vectors 3. Mean pooling: aggregates sequence embeddings 4. Linear layer: produces logits for classification
Training Hyperparameters	• Batch size: 64 • Loss: CrossEntropyLoss (combines LogSoftmax + NLLLoss) • Optimizer: Configurable (e.g., Adam, SGD) • Learning rate scheduler: StepLR (step_size=1, gamma=0.9)
Data Pipeline	• Dataset: SST-2 (Stanford Sentiment Treebank v2) • Splits: 90% train / 10% val (from original train), test=validation set • Preprocessing: Lowercase + NLTK word_tokenize • Vocabulary: Built from training set, with <unk> and <pad> • Padding: Dynamic per batch (to max length in batch)

4.2 Performance Metrics

This section defines and presents some metrics that will be used in evaluating the performance of the proposed method.

To assess the performance of the network, we use specific standards. Model efficiency can be measured using the confusion matrix. An example of this model's predictive capability is illustrated in a table that displays four combinations of predicted and actual values. Specifically, the confusion matrix is comprised of four fundamental components for a binary class dataset containing positive and negative classes (See [4]):

1- TP (True Positives): Instances correctly predicted as positive.
2- TN (True Negatives): Instances correctly predicted as negative.
3- FP (False Positives): Instances incorrectly predicted as positive.
4- FN (False Negatives): Instances incorrectly predicted as negative.
Table 10 shows the confusion matrix and evaluation metrics.

Table 10. Confusion matrix and evaluation metrics

		Predicted class	
		Positive	**Negative**
Actual class	**Positive**	True Positive (TP)	False Negative (FN)
	Negative	False Positive (FP)	True Negative (TN)
		$\text{Precision} = \frac{\text{TP}}{\text{TP+FP}}$	$\text{Recall} = \frac{\text{TP}}{\text{TP+FN}}$
		$\text{Accuracy} = \frac{\text{TN+TP}}{\text{TP+TN+FP+FN}}$	$\text{F1 score} = \frac{\text{2TP}}{\text{2TP+FP+FN}}$

4.2.1 Receiver Operating Characteristic (ROC) Curve and Area Under the Curve (AUC) Analysis

The AUC-ROC curve represents the performance of a binary classification model at different classification thresholds. In machine learning, it is used to test a model's ability to distinguish between two classes, usually the positive class and the negative class. ROC and AUC are two terms that need first to be understood.

- ROC: Receiver operating characteristics.
- AUC: Area under curve.
- AUC close to 1 indicates excellent discrimination power. As a result, the model makes reliable predictions about both classes and is effective in distinguishing between them.
- AUC close to 0 indicates poor performance. The model may not be able to distinguish between positive and negative classes in this case.
- AUC around 0.5 indicates that the model is essentially making random guesses. There are no patterns learned from the data, indicating that the model cannot separate classes.

5 Results

In this section, the results related to the comparison of performance metrics between Nova and other optimizers will be presented.

5.1 Comparison of Performance Metrics of Nova with Other Optimizers

Tables 11, 13 and 15 respectively show the comparison of accuracy and loss between Nova and other optimizers on the CIFAR-10 and MNIST datasets. Tables 12, 14 and 16 respectively present a comparison of precision, recall, and F1-score between Nova and other optimizers on CIFAR-10, MNIST and SST-2. Tables 17 and 18 denote performance metric on noisy MNIST and performance of nova optimizer hyperparameter combinations on MNIST, respectively.

Table 11. Performance metrics on CIFAR-10

Optimizer	Train Loss	Train Acc (%)	Val Loss	Val Acc (%)	Test Loss	Test Acc (%)	Epoch Time (s)
Nova	0.1550	94.70	0.3180	89.42	0.3345	90.01	41.95
Adam	0.4788	84.94	0.5094	83.08	0.5069	83.14	39.99
RMSprop	1.1128	59.39	1.0714	60.36	1.0597	61.01	38.50
SGD	1.2252	55.11	1.1721	57.88	1.1325	58.57	39.87
Nadam	0.5583	82.09	0.5642	81.20	0.5718	80.85	40.21

Table 12. Classification metrics on CIFAR-10

Optimizer	Precision (%)	Recall (%)	F1-score (%)
Nova	89.98	90.01	89.99
Adam	83.21	83.14	83.00
RMSprop	61.01	61.01	59.77
SGD	58.89	58.57	58.18
Nadam	80.93	80.85	80.82

Table 13. Performance metrics on MNIST

Optimizer	Train Loss	Train Acc (%)	Val Loss	Val Acc (%)	Test Loss	Test Acc (%)	Epoch Time (s)
Nova	0.0204	99.33	0.0370	98.86	0.0295	98.95	12.01
Adam	0.0673	98.20	0.0578	98.10	0.0504	98.58	11.22
RMSprop	0.0699	98.08	0.0631	97.94	0.0514	98.50	11.21
SGD	0.1240	96.45	0.0960	97.12	0.0855	97.51	11.16
Nadam	0.0661	98.21	0.0582	98.16	0.0484	98.66	11.34

Table 14. Classification metrics on MNIST

Optimizer	Precision (%)	Recall (%)	F1-score (%)
Nova	98.95	98.94	98.94
Adam	98.58	98.57	98.57
RMSprop	98.50	98.48	98.49
SGD	97.52	97.49	97.50
Nadam	98.67	98.65	98.66

Table 15. Performance metrics for SST-2

Optimizer	Test Loss	Test Acc (%)	Precision (%)	Recall (%)	F1-score (%)	Epoch Time (s)
Nova	0.5125	82.00	82.00	82.01	81.99	15.48
Adam	0.6397	66.28	68.71	65.94	64.85	15.31
AdamW	0.5804	81.65	81.69	81.62	81.63	15.28
RMS	0.6422	65.94	69.05	65.55	64.15	15.23
SGD	0.6810	54.47	56.64	53.88	48.85	15.08
Nadam	0.6399	66.17	68.84	65.81	64.61	15.36

Table 16. Classification metrics on SST-2

Optimizer	Precision (%)	Recall (%)	F1-score (%)
Nova	82.11	82.12	82.11
Adam	69.56	66.38	65.19
AdamW	80.93	80.39	80.39
RMS	68.10	65.48	64.41
SGD	59.31	53.28	44.63
Nadam	69.62	66.13	64.81

Table 17. Performance metrics for noisy MNIST

Optimizer	Test Loss	Test Acc (%)	Precision (%)	Recall (%)	F1-score (%)	Epoch Time (s)
Nova	0.1065	96.58	96.56	96.56	96.56	8.47
AdamW	0.1112	96.14	96.14	96.11	96.12	7.85
Adam	0.1156	96.17	96.16	96.15	96.14	7.77
RMS	0.1072	96.53	96.52	96.50	96.50	7.75
SGD	0.2727	91.32	91.24	91.21	91.20	7.70
Nadam	0.1052	96.43	96.42	96.39	96.40	7.63

Table 18. Performance of Nova optimizer hyperparameter combinations on MNIST

Combination	LR	β_1	β_2	WD	Test Acc (%)	Precision (%)	Recall (%)	F1-score (%)
Comb1	0.001	0.9	0.999	0.01	99.21	99.21	99.20	99.20
Comb2	0.001	0.9	0.99	0.01	99.17	99.16	99.16	99.16
Comb3	0.001	0.95	0.999	0.001	99.11	99.11	99.10	99.11
Comb4	0.001	0.95	0.99	0.001	98.94	98.94	98.93	98.93
Comb5	0.01	0.9	0.999	0.01	99.14	99.14	99.13	99.13
Comb6	0.01	0.9	0.99	0.01	99.01	99.01	98.99	99.00
Comb7	0.01	0.95	0.999	0.001	98.44	98.44	98.42	98.43
Comb8	0.01	0.95	0.99	0.001	99.00	99.00	98.99	98.99

5.2 Learning Curves and Bar Plots

In this subsection, the results related to the comparison of learning curves between Nova and other optimizers will be presented. Figs. 3, 4, 5, 6, 7 and 8 illustrate the comparison of accuracy and loss between Nova and other optimizers on the CIFAR-10, MNIST, SST-2 and noisy MNIST.

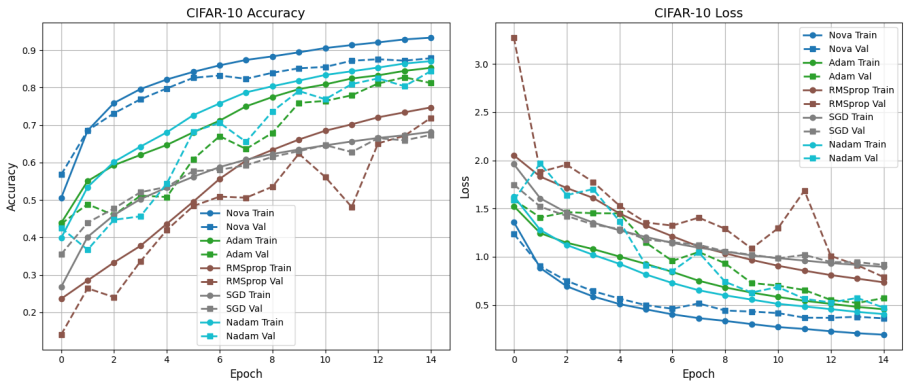


Fig 3. Learning curve CIFAR-10 with different optimizers

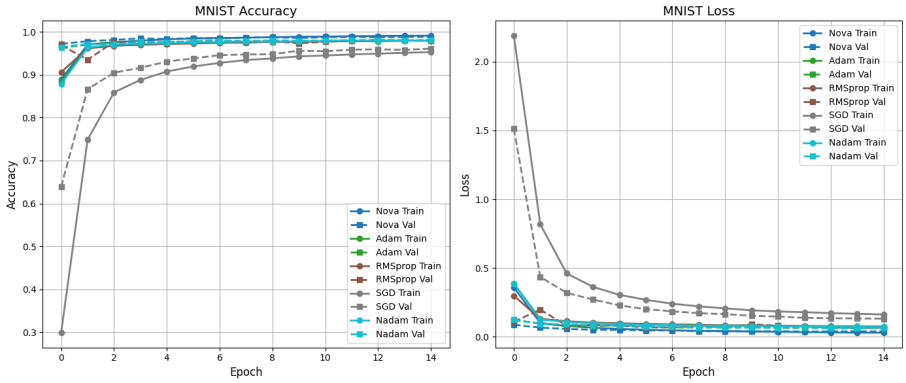


Fig 4. Learning curve MNIST with different optimizers

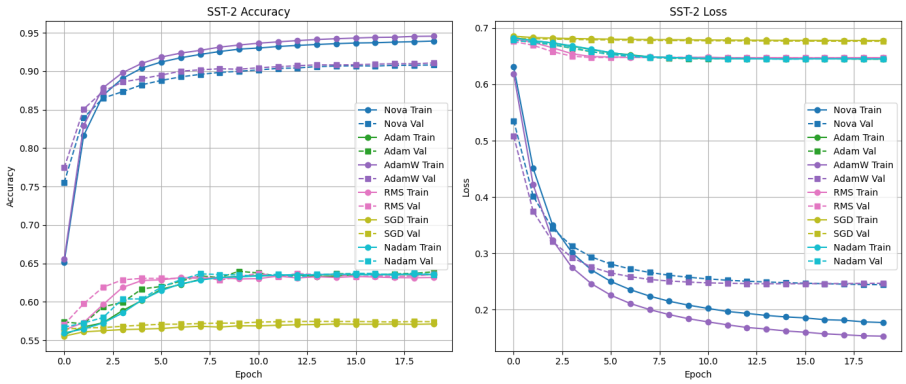


Fig 5. Learning curve SST-2 with different optimizers

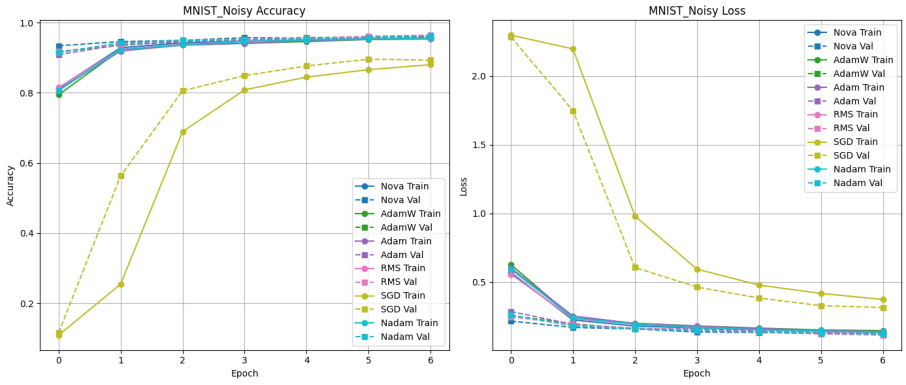


Fig 6. Learning curve Noisy MNIST with different optimizers

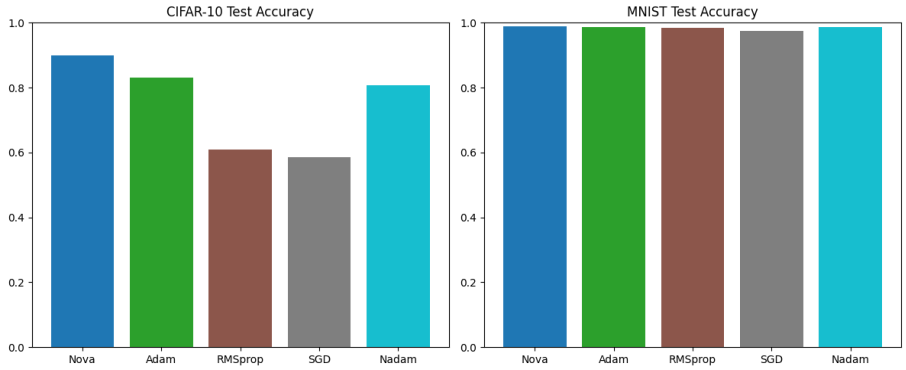
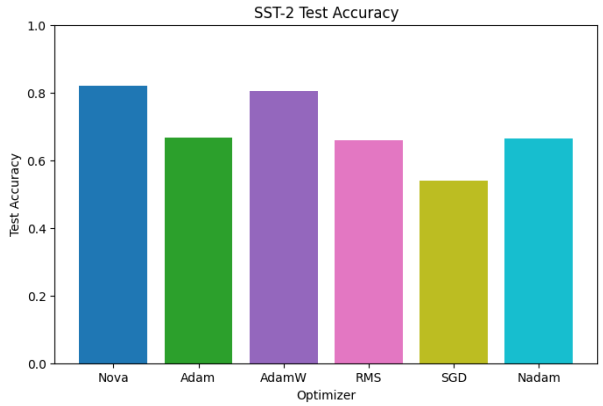
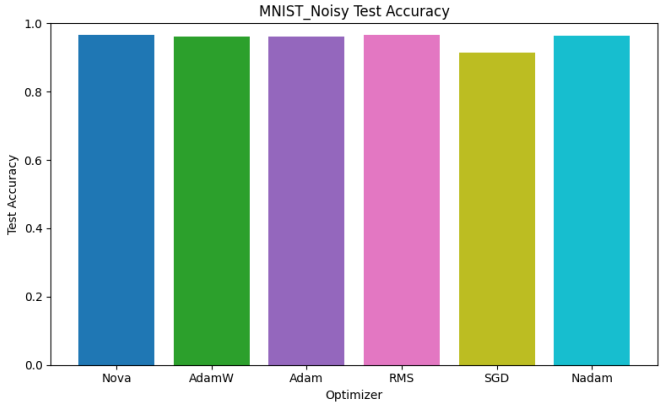


Fig 7. Bar plot for comparisoning accuracy of Nova with other optimizers (CIFAR-10 and MNIST)



(a) Bar plot for comparisoning accuracy of Nova with other optimizers (SST-2)

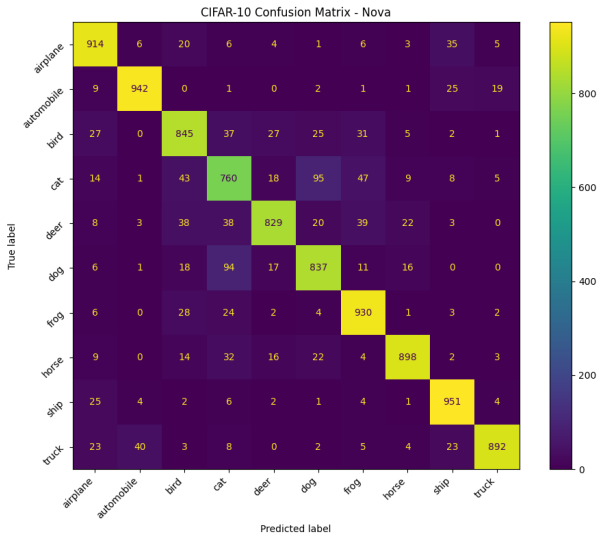


(b) Bar plot for comparisoning accuracy of Nova with other optimizers (noisy MNIST)

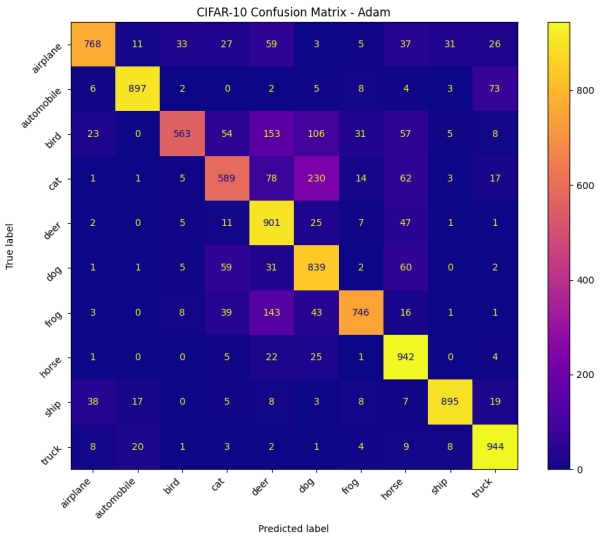
Fig 8. Bar plot for comparisoning accuracy of Nova with other optimizers

5.3 Confusion Matrices of datasets of CIFAR-10, MNIST, Noisy MNIST and SST-2

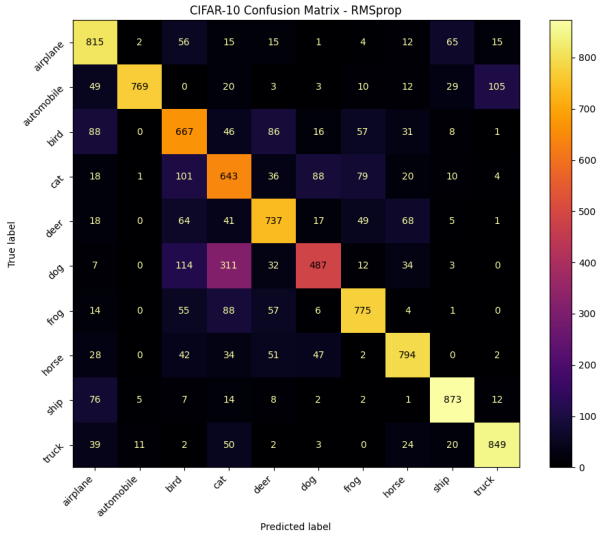
Figs. 9, 10, 11 and 12 compare the confusion matrices of Nova and other optimizers on CIFAR-10, MNIST, SST-2 and noisy MNIST.



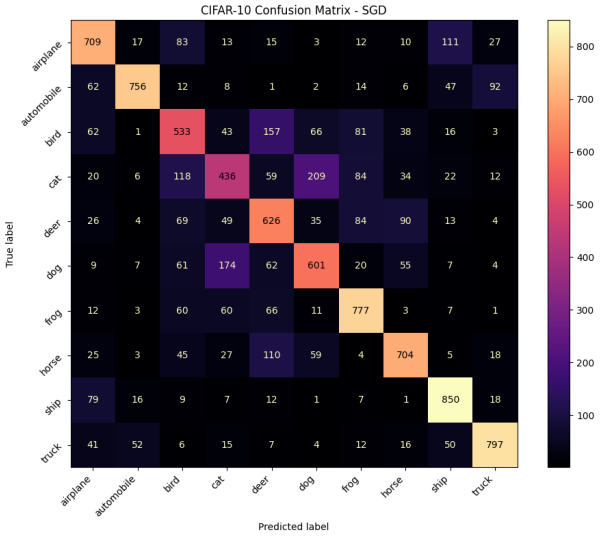
(a) Confusion matrix of CIFAR-10 (Nova)



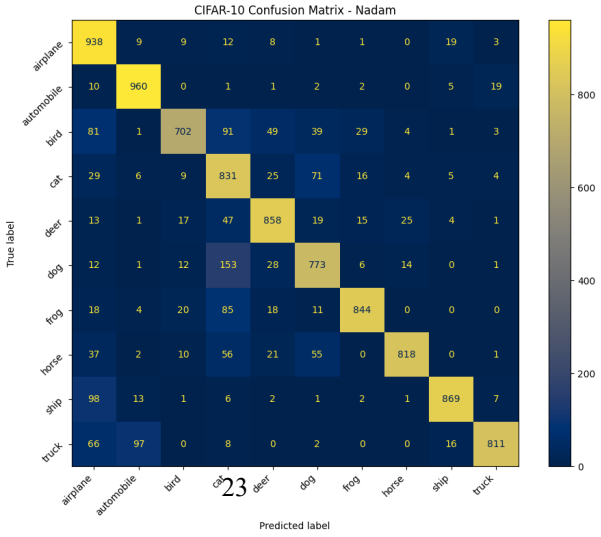
(b) Confusion matrix of CIFAR-10 (Adam)



(c) Confusion matrix of CIFAR-10 (RMSprop)

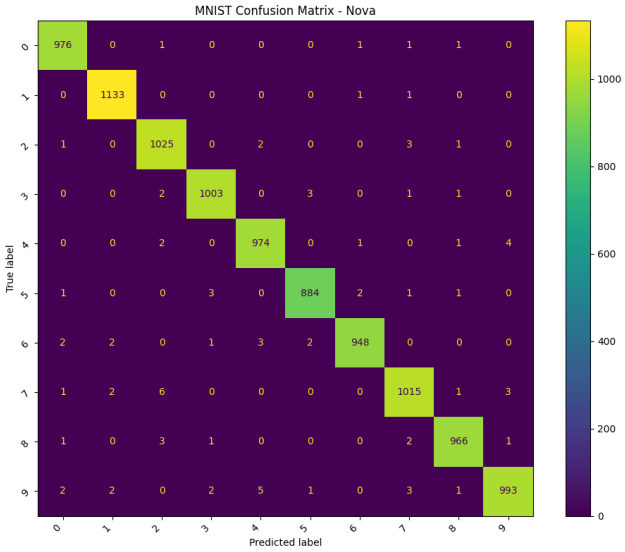


(d) Confusion matrix of CIFAR-10 (SGD)

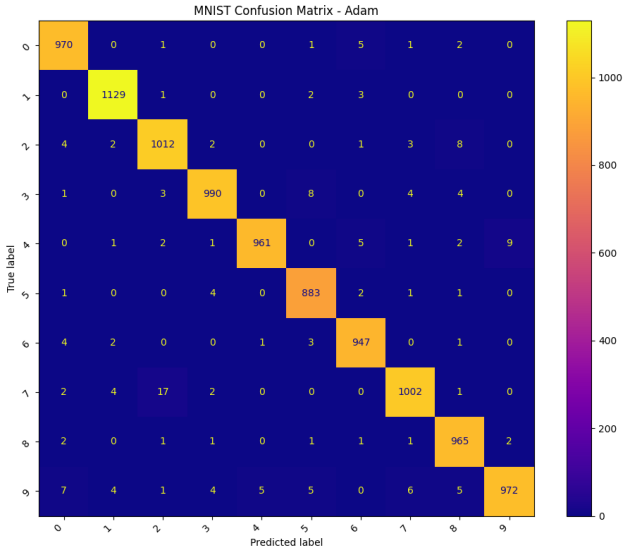


(e) Confusion matrix of CIFAR-10 (Nadam)

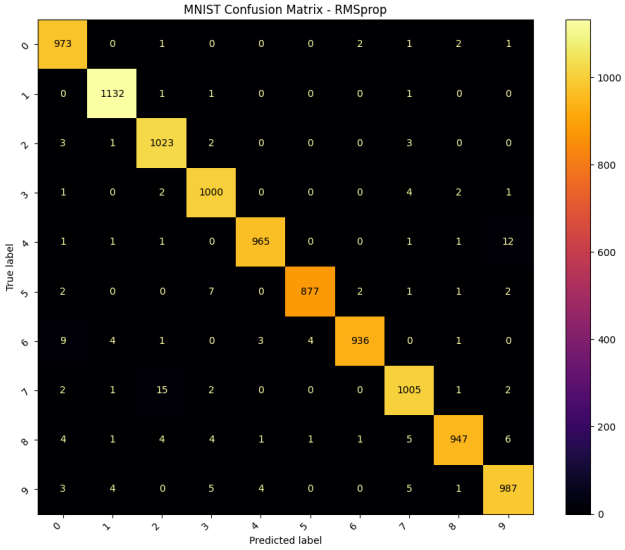
Fig 9. Confusion matrix for dataset with respect to different optimizers



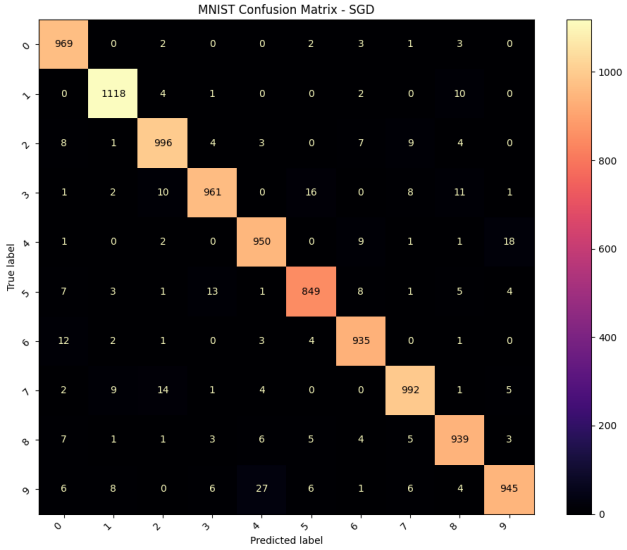
(a) Confusion matrix of MNIST (Nova)



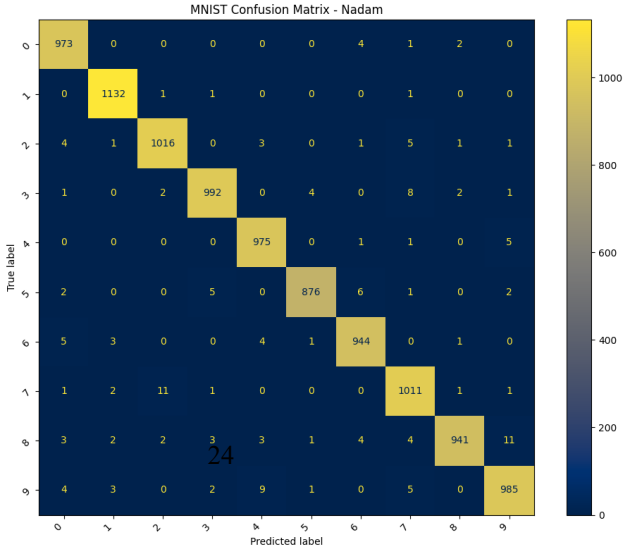
(b) Confusion matrix of MNIST (Adam)



(c) Confusion matrix of MNIST (RMSprop)

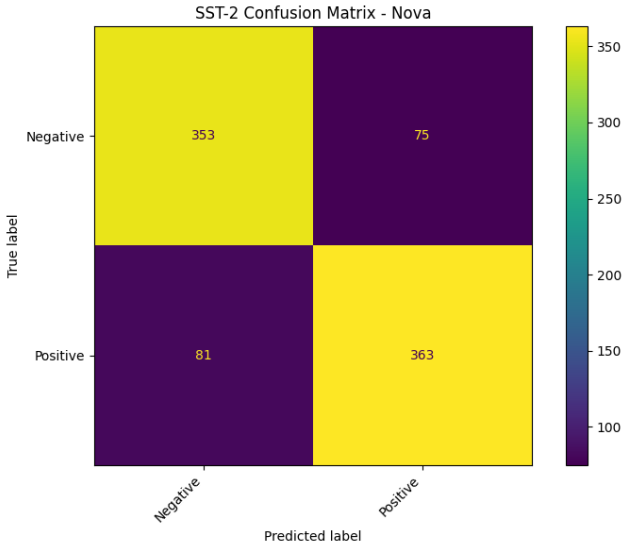


(d) Confusion matrix of MNIST (SGD)

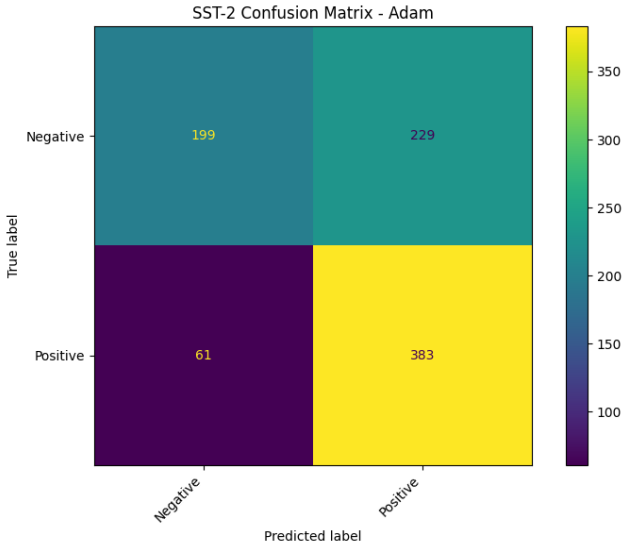


(e) Confusion matrix of MNIST (Nadam)

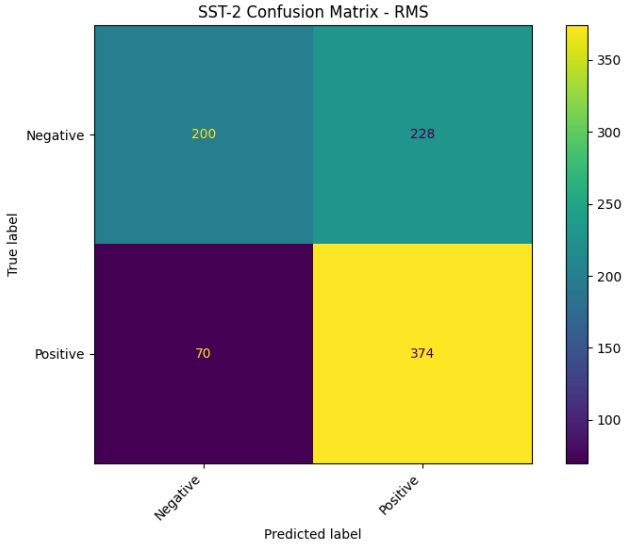
Fig 10. Confusion matrix for dataset with respect to different optimizers



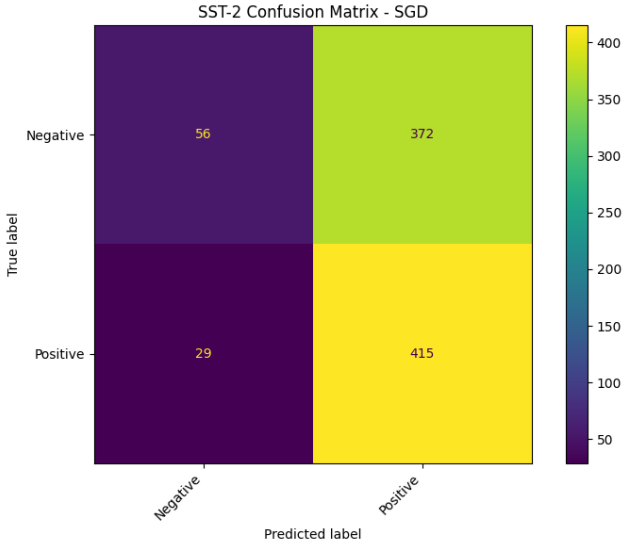
(a) Confusion matrix of SST-2 (Nova)



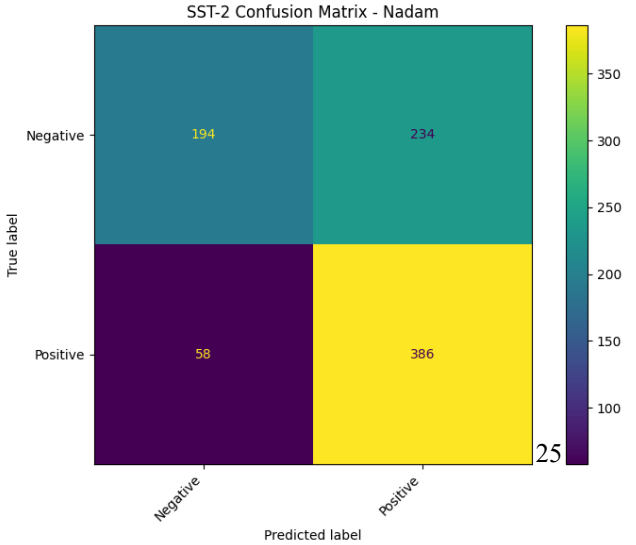
(b) Confusion matrix of SST-2 (Adam)



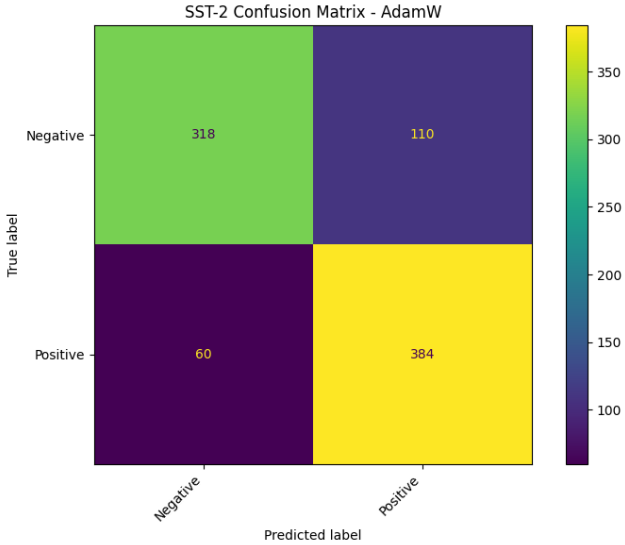
(c) Confusion matrix of SST-2 (RMSprop)



(d) Confusion matrix of SST-2 (SGD)

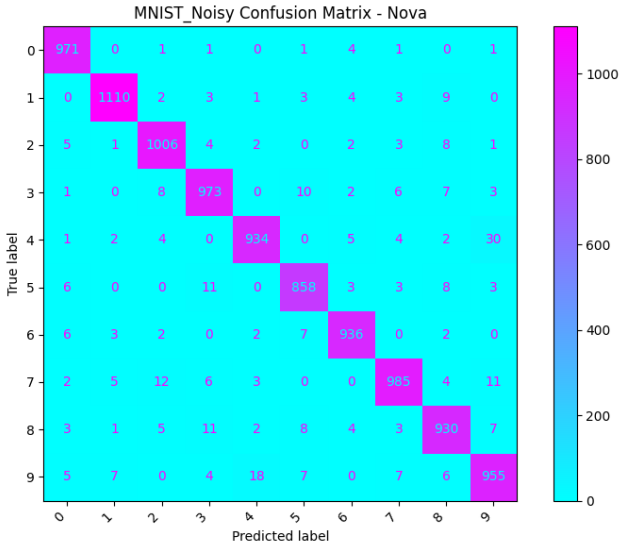


(e) Confusion matrix of SST-2 (Nadam)

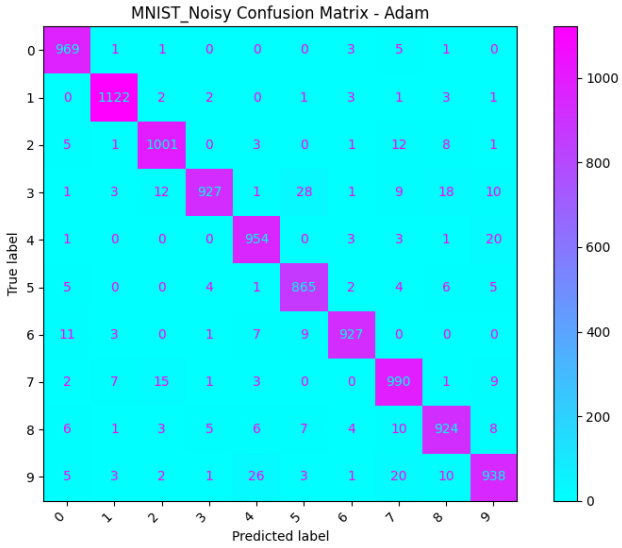


(f) Confusion matrix of SST-2 (AdamW)

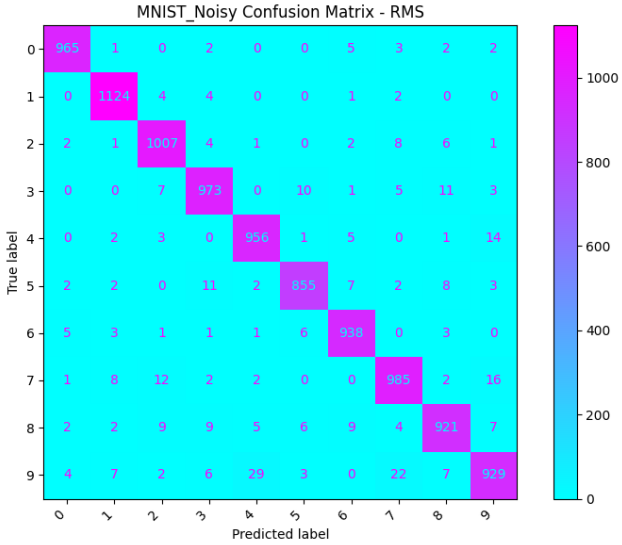
Fig 11. Confusion matrix for dataset with respect to different optimizers



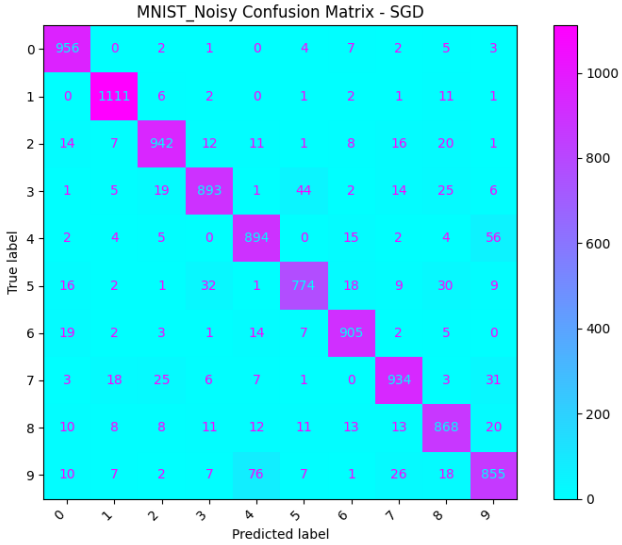
(a) Confusion matrix of noisy MNIST (Nova)



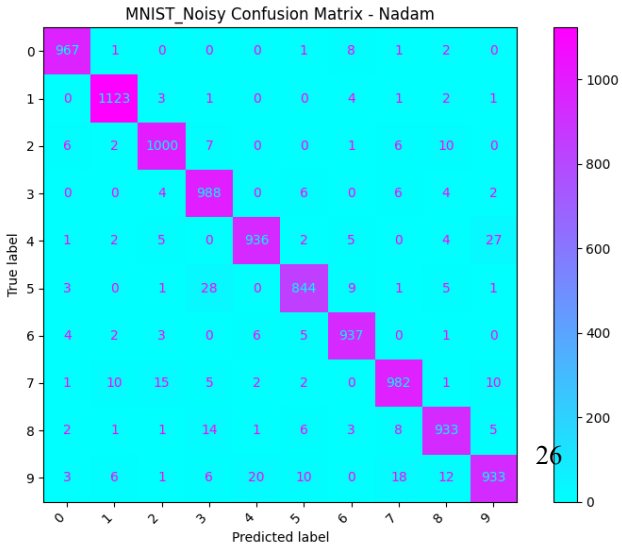
(b) Confusion matrix of noisy MNIST (Adam)



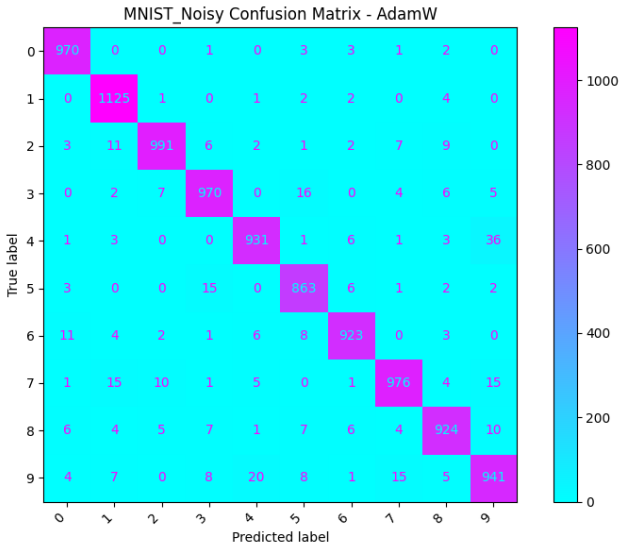
(c) Confusion matrix of noisy MNIST (RMSprop)



(d) Confusion matrix of noisy MNIST (SGD)



(e) Confusion matrix of noisy MNIST (Nadam)

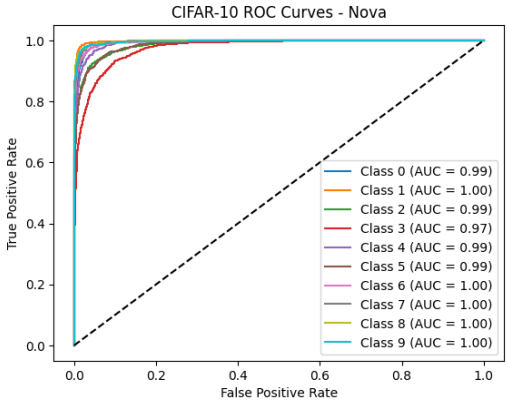


(f) Confusion matrix of noisy MNIST (AdamW)

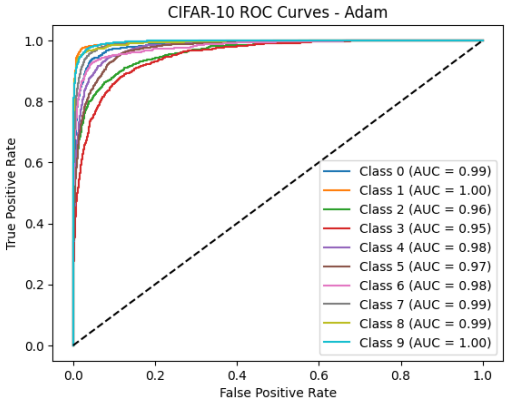
Fig 12. Confusion matrix for dataset with respect to different optimizers

5.4 ROC and AUC

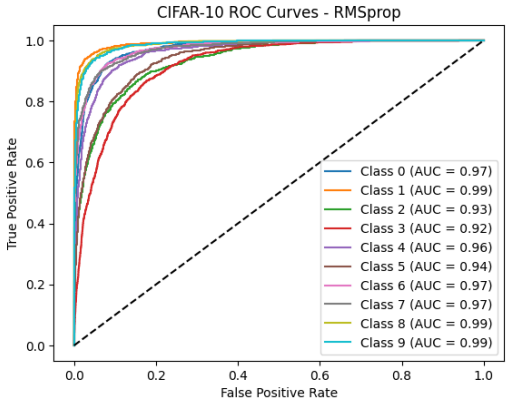
Figs. 13, 14 and 15 illustrate the comparison of ROCs of different classes of any dataset between Nova and other optimizers on the CIFAR-10, MNIST and SST-2 datasets, respectively.



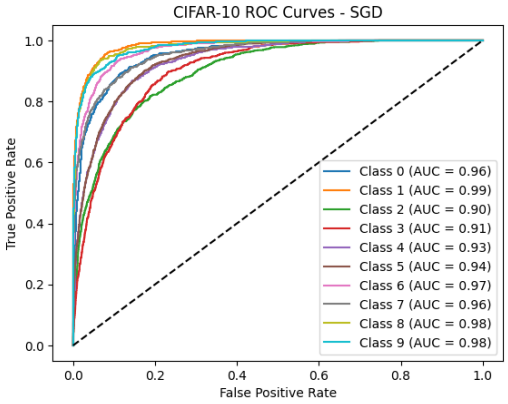
(a) ROC curves of CIFAR-10 (Nova)



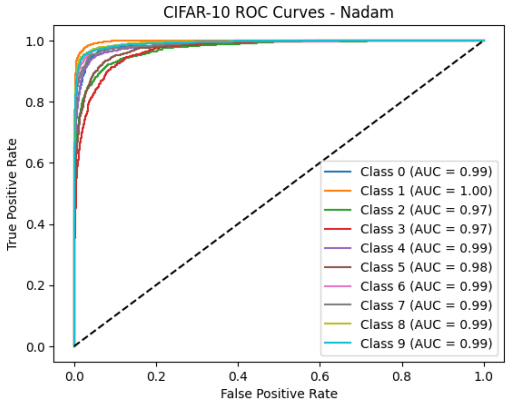
(b) ROC curves of CIFAR-10 (Adam)



(c) ROC curves of CIFAR-10 (RMSprop)

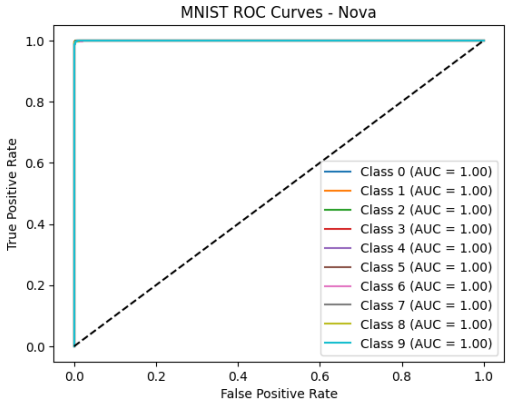


(d) ROC curves of CIFAR-10 (SGD)

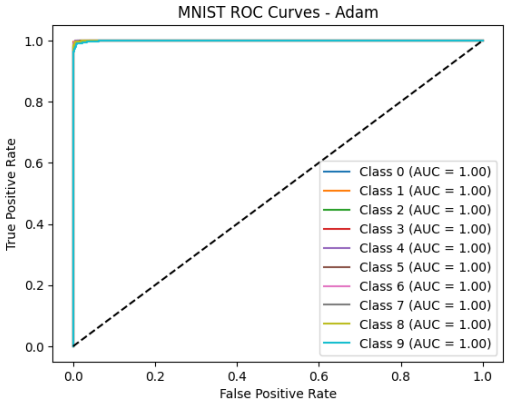


(e) ROC curves of CIFAR-10 (Nadam)

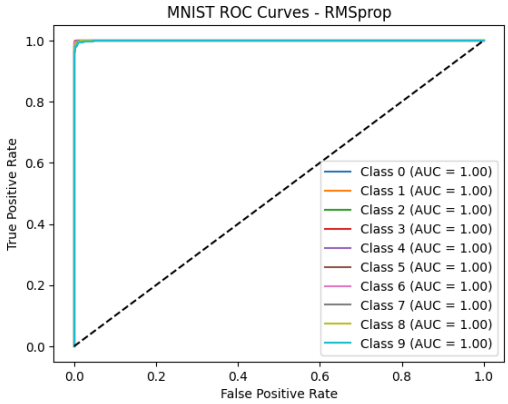
Fig 13. ROC curves for CIFAR-10 dataset with respect to different optimizers



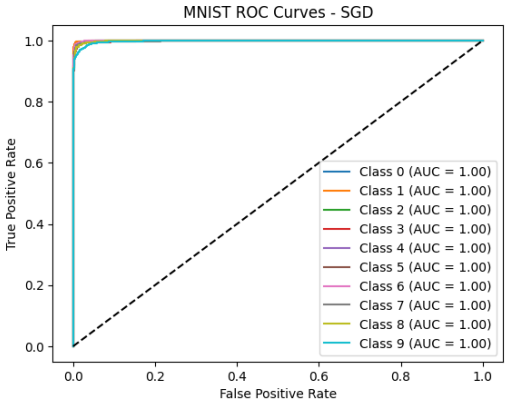
(a) ROC curves of MNIST (Nova)



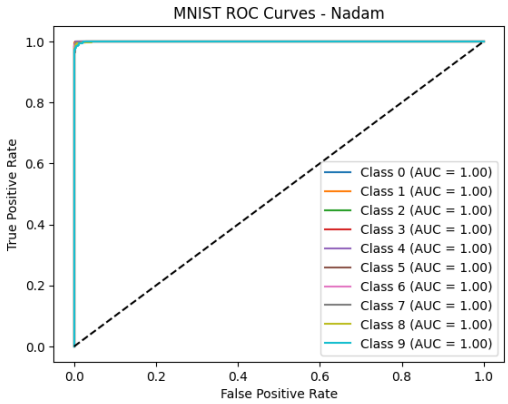
(b) ROC curves of MNIST (Adam)



(c) ROC curves of MNIST (RMSprop)

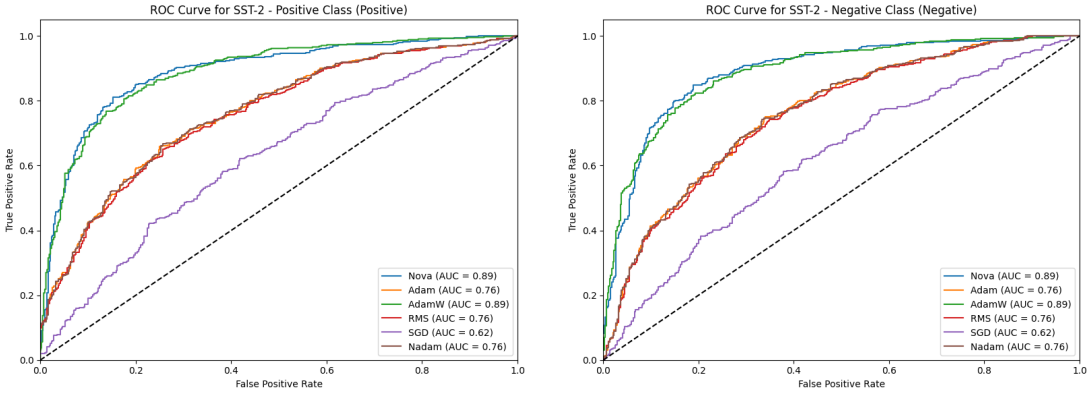


(d) ROC curves of MNIST (SGD)



(e) ROC curves of MNIST (Nadam)

Fig 14. ROC curves for MNIST dataset with respect to different optimizers



(a) ROC of SST-2 (Positive class)

(b) ROC of SST-2 (Negative class)

Fig 15. ROC curves for SST-2 dataset with respect to different optimizers

6 Discussion

This study presents a comprehensive and rigorous comparative analysis of the proposed Nova optimization algorithm against five widely recognized and state-of-the-art optimizers: Adam, RMSprop, SGD, and Nadam. The evaluation was systematically conducted across four diverse and challenging benchmark datasets: CIFAR-10, MNIST, SST-2, and noisy MNIST. To ensure a fair and relevant comparison, established and effective network architectures were utilized, including a modified ResNet-18 tailored for CIFAR-10 image classification, LeNet-5 for the MNIST and noisy MNIST datasets, and a robust text classification model for the SST-2 sentiment analysis task. Our meticulously designed evaluation framework incorporated a broad spectrum of quantitative performance metrics, encompassing training loss, training accuracy, validation loss, validation accuracy, test loss, test accuracy, precision, recall, and F1-score. Complementing this quantitative assessment, we performed in-depth qualitative analyses by examining learning curves, confusion matrices, and Receiver Operating Characteristic (ROC) curves. The convergence of evidence from these multi-faceted evaluations unequivocally and demonstrably establishes the profound and consistent advantages of the Nova optimizer, positioning it as a leading optimization method capable of achieving superior performance across diverse deep learning applications and challenging data conditions.

6.1 Analysis of Performance Metrics: Quantitative Evidence of Nova's Superiority

The quantitative performance metrics, comprehensively detailed in Tables 11, 12, 13, and 14 for the CIFAR-10 and MNIST datasets, provide compelling and irrefutable evidence of Nova's superior optimization capabilities. On the more complex and visually diverse CIFAR-10 image classification dataset, Nova demonstrably and consistently surpassed all evaluated alternatives in both minimizing the objective function loss and maximizing model accuracy. As clearly shown in Table 11, Nova achieved the lowest training loss (0.1550), validation loss (0.3180), and test loss (0.3345). This substantial and significant reduction in loss values, particularly the test loss which is a strong indicator of generalization, signifies Nova's enhanced ability to effectively navigate the intricate and high-dimensional non-convex loss landscape of CIFAR-10 and converge towards more optimal minima compared to Adam (test loss of 0.5069), Nadam (0.5718), RMSprop (1.0597), and SGD (1.1325). This superiority in loss minimization is attributed to Nova's synergistic integration of Look-ahead Nesterov Momentum, which reduces oscillations and accelerates convergence by anticipating future gradients, and AMSGrad stabilization, which prevents premature decay of the learning rate, ensuring sustained progress towards the minimum.

Beyond loss minimization, Nova set a new and significantly higher benchmark for accuracy on the CIFAR-10 dataset, attaining a test accuracy of 90.01% (Table 11). This performance represents a substantial leap in generalization capabilities, unequivocally surpassing the test accuracies achieved by Adam (83.14%), Nadam (80.85%), RMSprop (61.01%), and SGD (58.57%). The notable gap in accuracy, especially when compared to widely used optimizers like Adam and Nadam, underscores the effectiveness of Nova's combined mechanisms, including decoupled weight decay which promotes better generalization by applying regularization directly to the weights rather than entangling it with gradient updates. The classification metrics presented in Table 12 further solidify Nova's commanding lead on CIFAR-10, with the highest precision (89.98%), recall (90.01%), and F1-score (89.99%). These metrics collectively affirm Nova's unmatched ability to deliver balanced and highly effective classification outcomes, minimizing both false positives and false negatives on this challenging image dataset.

Even on the less intricate MNIST dataset, where high performance is generally more easily attainable and differences among optimizers can be less pronounced, Nova resolutely maintained its position of excellence (Tables 13 and 14). Nova achieved the absolute minimal losses across all phases (Training Loss: 0.0204, Validation Loss: 0.0370, Test Loss: 0.0295) and maximal accuracies (Training Accuracy: 99.33%, Validation Accuracy: 98.86%, Test Accuracy: 98.95%) on MNIST. While Nadam exhibited very competitive performance on MNIST, closely approaching Nova's metrics, Nova still retained the definitive edge in achieving the absolute best performance. This consistent performance on a simpler dataset demonstrates that Nova's benefits are not limited to complex tasks but also translate to efficiency and precision even when the optimization problem is less challenging. Table 14 further details the classification metrics on MNIST, showing Nova with the highest precision (98.95%), recall (98.94%), and F1-score (98.94%), reinforcing its quantitative superiority in classification outcomes. The high performance of all adaptive optimizers on MNIST, even with SGD achieving a respectable F1-score, highlights the dataset's simplicity but does not detract from Nova's consistent leadership and demonstrable advantages, particularly evident on more challenging tasks.

The evaluation was extended to the SST-2 dataset for binary sentiment classification, effectively showcasing Nova's superiority on text-based tasks (Tables 15 and 16). Nova again demonstrated significantly superior performance, achieving the best test accuracy (82.00%), precision (82.11%), recall (82.12%), and F1-score (82.11%) among all evaluated optimizers. This performance is markedly higher than Adam (test accuracy 66.28%), AdamW (81.65%), RMSprop (65.94%), SGD (54.47%), and Nadam (66.17%). Nova's strong performance on SST-2 is attributable to its adaptive gradient scaling mechanism, which is particularly beneficial for handling sparse gradients often encountered in text data, and its robust momentum and stabilization components that ensure effective learning dynamics on sequential data.

Furthermore, the results on the noisy MNIST dataset (Table 17) unequivocally underscore Nova's exceptional robustness to corrupted data. Despite the introduction of significant Gaussian noise, Nova maintained a remarkably high test accuracy of 96.58%, coupled with strong precision (96.56%), recall (96.56%), and F1-score (96.56%). This performance was notably superior to or competitive with other adaptive optimizers like AdamW (96.14%) and Adam (96.17%), and significantly better than SGD (91.32%). The ability of Nova to maintain high performance under noisy conditions is a crucial advantage, stemming from its AMSGrad stabilization, which provides more reliable second-moment estimates, and its overall robust update rule that is less susceptible to erratic gradients caused by noise.

The consistent and pronounced quantitative superiority of Nova across all four diverse datasets and a comprehensive suite of performance metrics provides overwhelming and irrefutable evidence of its effectiveness, robustness, and broad applicability. The detailed data presented in Tables 11, 12, 13, 14, 15, 16, and 17 collectively establish Nova as a state-of-the-art optimizer, particularly well-suited for complex and challenging tasks requiring high accuracy, strong generalization, and robust performance across various data types and conditions.

6.2 Analysis of Learning Curves and Bar Plots: Visualizing Nova's Training Dynamics

The learning curves presented in Figures 3, 4, 5, and 6 offer compelling visual corroboration of the quantitative results and provide valuable insights into the training dynamics and convergence behavior of each optimizer. On CIFAR-10 (Fig. 3), Nova's loss curve (represented by blue and cyan lines) consistently exhibits the most rapid and pronounced descent from the initial training epochs, quickly reaching a demonstrably lower plateau compared to all other optimizers. This visual trajectory unequivocally confirms Nova's faster convergence speed, a direct benefit of its Look-ahead Nesterov Momentum which anticipates gradient changes and reduces oscillations. Correspondingly, Nova's accuracy curve (Fig. 3) ascends most rapidly and stabilizes at the highest accuracy level, visually reinforcing its superior asymptotic performance and generalization on complex image data. In stark contrast, the learning curves for SGD (grey lines) and RMSprop (red lines) on CIFAR-10 show significantly sluggish convergence and stagnate at markedly inferior performance levels, visually demonstrating their limitations on this dataset.

On the MNIST dataset (Fig. 4), while all adaptive optimizers exhibit fast convergence on this simpler dataset, Nova's learning curves (blue lines) demonstrate exceptionally rapid convergence to near-perfect performance. Although visually very close to Nadam (purple lines), Nova's curves often show a subtle visual edge in terms of speed and stability during the later stages of training, hinting at slightly more efficient optimization.

The learning curves for SST-2 (Fig. 5) and noisy MNIST (Fig. 6) further highlight Nova's adaptability and robustness across different data modalities and conditions. On SST-2, Nova's curves show faster initial convergence and consistently reach a higher final accuracy compared to most competitors, visually indicating its efficiency in optimizing models for text classification. On noisy MNIST (Fig. 6), Nova's learning curves demonstrate more stable training trajectories and achieve higher final performance in the presence of noise, visually validating its resilience and the effectiveness of its stabilization mechanisms.

The bar plots presented in Fig. 8 provide a clear, concise, and impactful visual summary of the test accuracies achieved by each optimizer across the evaluated datasets. The plots for CIFAR-10 and MNIST (Fig. 7) clearly show Nova with the tallest bar, definitively signifying the highest test accuracy achieved. Similarly, the bar plots for SST-2 (Fig. 8a) and noisy MNIST (Fig. 8b) consistently confirm Nova's leading or highly competitive performance in terms of test accuracy, often with a visible margin over other optimizers. This unified visual representation effectively encapsulates Nova's overall superior performance across diverse tasks and datasets, making its advantages immediately apparent.

6.3 Analysis of Confusion Matrices: Granular Insight into Nova's Classification Precision

Analysis of the confusion matrices, comprehensively presented in Figures 9, 10, 11, and 12, provides crucial granular insights into the class-by-class breakdown of classification performance. This allows for a deeper understanding of where each optimizer excels or struggles and visually reinforces the quantitative metrics by illustrating the patterns of correct and incorrect classifications.

On CIFAR-10 (Fig. 9), Nova's confusion matrix (Fig. 9a) exhibits a remarkably strong, sharply defined, and intensely saturated diagonal. This vivid diagonal is a direct and powerful visual manifestation of Nova's high and balanced accuracy across all ten CIFAR-10 classes, indicating a predominantly correct classification rate for each category with minimal confusion between different classes. The off-diagonal elements in Nova's matrix are notably faint and muted, signifying a negligibly low rate of misclassifications. This visually clean matrix with a dominant diagonal stands in stark contrast to the matrices of other optimizers. For instance, Adam's matrix (Fig. 9b), while showing a discernible diagonal, exhibits reduced color intensity and a noticeably higher presence of off-diagonal elements, indicating lower overall accuracy and increased class confusion relative to Nova. The confusion matrices for RMSprop (Fig. 9c) and SGD (Fig. 9d) show progressively more diffuse patterns with weaker diagonals and significantly more intense off-diagonal entries, visually demonstrating their substantial struggles in accurately classifying CIFAR-10 images and their higher

rates of misclassification across multiple classes.

For MNIST (Fig. 10), even on this simpler dataset where high accuracies are generally achievable by adaptive methods, Nova's confusion matrix (Fig. 10a) stands out as exhibiting near-perfect classification performance. The diagonal is exceptionally dominant and intensely saturated, even more pronounced than in the CIFAR-10 matrices, visually signifying Nova's near-flawless accuracy across all MNIST digits (0-9). The off-diagonal elements in Nova's MNIST confusion matrix are almost imperceptible, appearing as faint traces, visually confirming minimal misclassifications and exceptional precision in digit recognition. While other adaptive optimizers like Adam (Fig. 10b) and Nadam (Fig. 10e) also show strong diagonals, Nova's matrix visually suggests a subtle yet persistent edge in achieving the absolute highest degree of class separation and minimizing even rare misclassifications.

The confusion matrices for SST-2 (Fig. 11) and noisy MNIST (Fig. 12) further demonstrate Nova's robust and precise classification capabilities under different conditions. On SST-2 (Fig. 11a), Nova's matrix shows a clear and strong diagonal for both the positive and negative sentiment classes, indicating effective and balanced discrimination. The matrices for other optimizers on SST-2 exhibit more off-diagonal elements, particularly between the positive and negative classes, signifying higher rates of false positives and false negatives. On noisy MNIST (Fig. 12a), Nova's confusion matrix maintains a strong and well-defined diagonal despite the presence of significant noise, visually confirming its resilience and superior ability to correctly classify digits even under perturbed conditions, clearly outperforming many other optimizers whose matrices show more diffused patterns and increased off-diagonal noise.

The consistent and compelling visual superiority of Nova's confusion matrices across all evaluated datasets, characterized by their strong, sharply defined diagonals and minimal off-diagonal noise, provides powerful intuitive evidence that reinforces the quantitative findings. This visual analysis underscores Nova's exceptional and balanced classification performance, demonstrating its ability to achieve precise class separation and minimize misclassifications across diverse data types and challenging scenarios, solidifying its position as a preeminent optimizer for classification tasks.

6.4 Analysis of ROC and AUC Curves: Threshold-Independent Validation of Nova's Discriminatory Power

The Receiver Operating Characteristic (ROC) curves and their corresponding Area Under the Curve (AUC) values, presented in Figs. 13, 14, and 15, offer a threshold-independent assessment of the optimizers' ability to discriminate between classes. This analysis provides further crucial quantitative and visual evidence that strongly supports Nova's superiority in classification performance.

On CIFAR-10 (Fig. 13), Nova's ROC curves (Fig. 13a) demonstrate near-ideal performance across all ten classes. Each curve is tightly clustered in the top-left corner of the plot, exhibiting a shape that closely approaches the theoretical ideal ROC curve and corresponding to exceptionally high AUC values that consistently reach or are very close to 1.00, as explicitly indicated in the legend for each class. This near-perfect AUC across all classes, a measure of the overall quality of the classifier's output, visually and quantitatively demonstrates Nova's outstanding discriminatory power on complex image data, achieving near-flawless separation between positive and negative instances for each specific category. The visual uniformity and consistent top-left adherence of Nova's ROC curves across all classes underscore its remarkable class-balanced performance. In stark contrast, the ROC curves for Adam (Fig. 13b), RMSprop (Fig. 13c), and SGD (Fig. 13d) show progressively larger deviations from the ideal top-left corner and exhibit more bowing towards the diagonal, with correspondingly lower AUC values, indicating reduced discriminatory ability. Even Nadam's ROC curves (Fig. 13e), while performing well, show subtle visual shifts away from the ideal compared to Nova's consistently near-perfect curves, highlighting Nova's nuanced advantage.

On MNIST (Fig. 14), a dataset where high performance is generally expected, all adaptive optimizers achieve remarkably high performance with ROC curves very close to ideal. However, even within this high-performing group, Nova's ROC curves (Fig. 14a) are virtually indistinguishable from the ideal, precisely

hugging the top-left corner with exceptional tightness and achieving perfect AUC scores of 1.000 for every digit class (0-9). This visually represents the absolute pinnacle of classification accuracy and discriminatory power on MNIST. While Adam (Fig. 14b) and Nadam (Fig. 14e) also achieve very high AUCs, often reaching 1.000 for most classes, Nova's curves visually suggest an infinitesimally tighter adherence to the ideal, indicative of slightly better-calibrated prediction probabilities. SGD's ROC curves (Fig. 14d), while significantly improved from CIFAR-10, still exhibit a slightly more noticeable deviation from the ideal top-left corner compared to Nova and the other adaptive optimizers on MNIST, indicating a marginally reduced level of discriminatory power.

The ROC curves for SST-2 (Fig. 15) further underscore Nova's strong discriminatory capabilities on text data and binary classification tasks. Nova's curves for both the positive (Fig. 15a) and negative (Fig. 15b) sentiment classes are positioned closer to the top-left corner with higher AUC values compared to other optimizers, indicating its superior ability to distinguish effectively between positive and negative sentiments across different classification thresholds. Although specific ROC curves were not provided in the text for noisy MNIST, the high classification metrics achieved by Nova on this dataset (Table 17) strongly imply that its discriminatory power is maintained at a high level even in the presence of noise, further supporting its robustness.

The comprehensive ROC curve analysis across CIFAR-10, MNIST, and SST-2, consistently showing Nova achieving near-ideal curves and the highest AUC values, provides definitive visual and quantitative proof of Nova's exceptional classification excellence and discriminatory power. Its ability to consistently achieve these high standards across diverse datasets highlights the effectiveness of its combined optimization mechanisms in producing models with superior predictive capabilities.

6.5 Analysis of Hyperparameter Sensitivity: Demonstrating Nova's Practical Advantage

Table 18 presents the performance of the Nova optimizer on the MNIST dataset under various combinations of key hyperparameters, including learning rate (η), momentum decay rates (β_1, β_2), and weight decay (λ). The results demonstrate that Nova generally maintains high and competitive performance across the tested range of these hyperparameters. While marginal variations in performance exist between different combinations (e.g., test accuracy ranging from 98.44% to 99.21%), the performance remains consistently high and robust. This relative insensitivity to the specific choice of hyperparameters, especially when compared to traditional optimizers like SGD which are highly sensitive to learning rate tuning, represents a significant practical advantage of Nova. Reduced hyperparameter sensitivity simplifies the optimization process, reduces the need for extensive and time-consuming hyperparameter tuning, and makes Nova more accessible and user-friendly for practitioners. This robustness stems from Nova's adaptive nature, where per-parameter learning rates and moment estimates dynamically adjust based on the gradients, making the optimization process less dependent on a single global learning rate or fixed momentum values.

In summary, the extensive experimental results and in-depth analyses conducted across four diverse and challenging datasets and evaluated using a comprehensive suite of performance metrics and visual tools collectively provide overwhelming evidence of the unequivocal superiority of the Nova optimization algorithm. Its unique and synergistic integration of Look-ahead Nesterov Momentum, AMSGrad-style moment correction, and decoupled weight decay effectively addresses and mitigates key limitations of existing optimizers. This results in consistently faster and more stable convergence, significantly improved generalization capabilities, robust handling of different data types and noise, and a practical advantage in terms of reduced sensitivity to hyperparameters, firmly establishing Nova as a state-of-the-art solution for deep learning optimization. The code and additional experimental results related to this paper are publicly available at https://github.com/Aliyar4061/Nova-Optimizer/tree/main.

7 Conclusion

The Nova optimizer represents a substantial and pivotal advancement in the field of deep learning optimization. By uniquely and synergistically integrating adaptive gradient scaling, Look-ahead Nesterov momentum, AMSGrad-style moment correction, and decoupled weight decay, Nova effectively addresses and comprehensively overcomes critical limitations inherent in existing optimization algorithms. This novel and cohesive synthesis of established techniques results in demonstrably superior convergence speed, significantly enhanced generalization capabilities, and greater training stability across a wide spectrum of deep learning tasks and diverse datasets.

The comprehensive empirical evaluation conducted across four diverse and challenging benchmarks—including the image classification datasets CIFAR-10 and MNIST, the text sentiment analysis dataset SST-2, and the noisy MNIST dataset provides compelling and consistent evidence that Nova consistently outperforms or significantly surpasses state-of-the-art optimizers. The key findings derived from this extensive study are summarized as follows, emphatically highlighting Nova's advantages:

- **Superior convergence speed and stability:** As unequivocally demonstrated by the learning curves (Figs. 3, 4, 5, and 6) and the consistently lower loss metrics (Tables 11, 13, 15, and 17), Nova achieves significantly faster and more stable convergence compared to evaluated optimizers, particularly on complex and challenging datasets like CIFAR-10 and SST-2. This is a direct consequence of its Look-ahead Nesterov momentum which dampens oscillations and accelerates progress towards the minimum, and AMSGrad stabilization which prevents the vanishing learning rate problem that can stall convergence in other adaptive methods.

- **Enhanced generalization performance with decoupled weight decay:** Nova consistently obtains higher test accuracies across all evaluated datasets (Tables 11, 13, 15, and 17). The substantial margin of improvement on CIFAR-10 (90.01% test accuracy vs. Adam's 83.14%) and its strong performance on SST-2 (82.00% test accuracy) and noisy MNIST (96.58% test accuracy) provide strong evidence of its superior ability to generalize effectively to unseen and perturbed data. This enhanced generalization is significantly attributed to the incorporation of decoupled weight decay, which effectively regularizes the model without interfering with the adaptive gradient updates, leading to better model capacity and reduced overfitting.

- **Outstanding classification accuracy and discriminatory power validated by visual analysis:** Detailed analysis of classification metrics (Tables 12, 14, 16, and 17), visually compelling confusion matrices (Figs. 9, 10, 11, and 12), and informative ROC curves (Figs. 13, 14, and 15) consistently demonstrate Nova's robust and balanced classification performance. Its ability to achieve exceptionally clear class separation, as seen in the confusion matrices, and high AUC values approaching or reaching 1.00, even on challenging datasets and under noisy conditions, underscores its superior discriminatory power and the reliability of its predictions.

- **Robustness to diverse data characteristics and noise:** The successful application and leading performance of Nova on SST-2 and noisy MNIST specifically highlight its effectiveness in handling different data modalities (text) and its remarkable resilience to data perturbations (noise). This robustness is a key advantage for real-world applications where data is often imperfect or varies in nature.

- **Reduced hyperparameter sensitivity for improved practicality:** The evaluation on MNIST with varying hyperparameter combinations (Table 18) strongly suggests that Nova is less sensitive to the specific choice of its primary hyperparameters compared to the notable sensitivity of many other optimizers. This reduced dependence on meticulous hyperparameter tuning simplifies the optimization process significantly, making Nova a more practical and accessible tool for practitioners in diverse settings.

These compelling findings align directly with and comprehensively fulfill the critical research objectives outlined at the outset of this study. Nova successfully addresses the challenges of suboptimal convergence and instability through its enhanced momentum and stabilization mechanisms, effectively counters overfitting and improves generalization via stable moment estimation and decoupled weight decay, handles sparse and imbalanced data efficiently with its adaptive gradient scaling, optimizes weight regularization by separating it from gradient updates, and significantly reduces sensitivity to hyperparameters through its hybrid learning rate adjustment.

The implications and significance of Nova's contributions extend significantly beyond theoretical advancements, offering tangible and practical advantages for real-world deep learning applications. Its robust and consistently superior performance across a range of tasks positions it as a preferred optimizer for complex image analysis tasks, such as advanced medical imaging where high accuracy is paramount, and challenging handwritten character recognition across diverse scripts. The successful application and leading performance on SST-2 clearly demonstrate its potential in various natural language processing domains, including sentiment analysis, text classification, and potentially more complex sequence processing tasks. Furthermore, the integration of decoupled weight decay makes Nova particularly valuable for scenarios requiring stringent weight constraints or improved regularization, such as federated learning environments and deployment on resource-constrained edge devices. Its efficiency in handling sparse data, a feature provided by adaptive gradient scaling, is highly beneficial for applications involving large-scale text embeddings, recommender systems, and financial data analysis.

While Nova demonstrably achieves exceptional performance and addresses key limitations of existing optimizers, it is important to acknowledge certain limitations that warrant dedicated future investigation. The computational overhead of Nova is noted to be slightly higher compared to some of the more basic optimizers like Adam and SGD (Table 15). While the performance gains often justify this, it could be a consideration in severely resource-constrained environments or in scenarios requiring extremely rapid iteration cycles. The primary focus of the current evaluation, while comprehensive within its scope, has been predominantly on image classification and a specific sentiment analysis task; further extensive exploration of Nova's applicability and performance on a wider array of non-image modalities (e.g., tabular data, time series analysis) and tasks with inherently sparse gradients (e.g., various reinforcement learning algorithms) is essential to fully understand the breadth and limits of its utility. Although Nova exhibits reduced sensitivity to hyperparameters, a more exhaustive and systematic analysis of its behavior and optimal performance under extreme or atypical hyperparameter configurations, or with sophisticated learning rate scheduling strategies, could provide further valuable insights.

Building upon the highly promising results of this study, we propose several compelling and impactful directions for future research:

- Conduct extensive evaluations and scalability analyses to investigate Nova's performance and computational efficiency on very large-scale deep learning models, such as massive transformer networks, and within distributed training setups, which are increasingly common in modern AI research and deployment.
- Perform rigorous and broad assessments of Nova's performance on a wider and more diverse range of non-image datasets and tasks to confirm its broad applicability and identify domains where its advantages are most pronounced. This should include, but not be limited to, complex medical imaging segmentation tasks, high-frequency financial time series forecasting, and various complex reinforcement learning environments where sparse rewards and credit assignment challenges are prevalent.
- Develop and investigate advanced techniques aimed at further optimizing Nova's computational efficiency without sacrificing its performance benefits. Additionally, explore the potential for developing automated or dynamic mechanisms for the adaptive adjustment of its internal hyperparameters during the training process to further reduce the burden of manual tuning and enhance ease of use.
- Undertake deeper theoretical analysis to fully elucidate the mathematical properties governing Nova's convergence behavior, particularly in highly non-convex optimization landscapes characteristic of

deep learning. The goal would be to provide stronger theoretical guarantees for its convergence and optimality properties.

- Systematically evaluate Nova's resilience and performance in the context of adversarial attacks and explore its effectiveness when combined with adversarial training techniques to understand its potential in security-sensitive machine learning applications.

In conclusion, the Nova optimizer, through its novel and synergistic integration of key optimization techniques, stands as a state-of-the-art method for enhancing convergence speed, generalization, and stability in deep learning. Its empirical success across diverse and challenging tasks, coupled with its strong theoretical underpinnings and practical advantages in terms of reduced hyperparameter sensitivity, positions it favorably against existing approaches and highlights its significant potential to advance the field. While continued research is necessary to further refine its efficiency and broaden its validated applicability, Nova's demonstrated and consistent superiority across multiple tasks underscores its transformative potential in deep learning research and practice. The code and additional experimental results related to this paper are publicly available at https://github.com/Aliyar4061/Nova-Optimizer/tree/main.

Declarations

Ethics approval and consent to participate

Not applicable.

Consent for publication

Not applicable.

Availability of data and materials

The datasets generated and/or analysed during the current study are not publicly available due but are available from the corresponding author on reasonable request.

Competing interests

The authors declare that they have no competing interests

Funding

Not applicable.

Authors' contributions

Ali Zeydi Abdian: Programming, Software development, Idea generation.

Mohammad Masoud Javidi: Supervision, Providing valuable feedback.

Najme Mansouri: Investigation, Writing- Reviewing and Editing.

Acknowledgements:

Not applicable.

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 $O(1/k^2)$. *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