*electronics*

MDPI

*Article*

# An Improvement of Adam Based on a Cyclic Exponential Decay Learning Rate and Gradient Norm Constraints

**Yichuan Shao [1], Jiapeng Yang [2], Wen Zhou [2], Haijing Sun [1,*], Lei Xing [3], Qian Zhao [4] and Le Zhang [1]**

[1] School of Intelligent Science & Engineering, Shenyang University, Shenyang 110044, China; shaoyichuan@syu.edu.cn (Y.S.); zhangle@syu.edu.cn (L.Z.)
[2] School of Information Engineering, Shenyang University, Shenyang 110044, China; yangjiapeng1024@outlook.com (J.Y.); zhouwend@outlook.com (W.Z.)
[3] School of Chemistry and Chemical Engineering, University of Surrey, Guildford GU2 7XH, UK; l.xing@surrey.ac.uk
[4] School of Science, Shenyang University of Technology, Shenyang 110044, China
* Correspondence: sunhaijing@syu.edu.cn

**Abstract:** Aiming at a series of limitations of the Adam algorithm, such as hyperparameter sensitivity and unstable convergence, in this paper, an improved optimization algorithm, the Cycle-Norm-Adam (CN-Adam) algorithm, is proposed. The algorithm integrates the ideas of a cyclic exponential decay learning rate (CEDLR) and gradient paradigm constraintsand accelerates the convergence speed of the Adam model and improves its generalization performance by dynamically adjusting the learning rate. In order to verify the effectiveness of the CN-Adam algorithm, we conducted extensive experimental studies. The CN-Adam algorithm achieved significant performance improvementsin both standard datasets. The experimental results show that the CN-Adam algorithm achieved 98.54% accuracy in the MNIST dataset and 72.10% in the CIFAR10 dataset. Due to the complexity and specificity of medical images, the algorithm was tested in a medical dataset and achieved an accuracy of 78.80%, which was better than the other algorithms. The experimental results show that the CN-Adam optimization algorithm provides an effective optimization strategy for improving model performance and promoting medical research.

**Keywords:** deep learning; cyclic exponential decay learning rate; gradient paradigm constraints; Adam algorithm; CN-Adam algorithm



**Citation:** Shao, Y.; Yang, J.; Zhou, W.; Sun, H.; Xing, L.; Zhao, Q.; Zhang, L. An Improvement of Adam Based on a Cyclic Exponential Decay Learning Rate and Gradient Norm Constraints. *Electronics* **2024**, *13*, 1778. https://doi.org/10.3390/electronics13091778

Academic Editor: Alberto Fernandez Hilario

Received: 11 April 2024
Revised: 26 April 2024
Accepted: 1 May 2024
Published: 4 May 2024



## 1. Introduction

With the rapid development of deep learning, the importance of optimization algorithms in training neural networks is increasing. Choosing an appropriate optimization algorithm is crucial for the training speed and performance of a model. In current deep learning practice, the Adam optimization algorithm is widely used, which has the advantages of fast convergence and good generalization performance. However, optimization algorithms like the Adam algorithm have some drawbacks, such as sensitivity to the learning rate and selection of hyperparameters. In order to solve this problem, researchers have started to improve the Adam algorithm regarding different aspects. To optimize the learning rate, Yiming Jiang et al. [1] proposed the UAdam algorithm, which introduces a generalized second-order momentum form. An increase in the parameter β leads to a decrease in the convergence neighborhood, and by adjusting β, the convergence performance of the algorithm can be controlled. Liu et al. [2] proposed the RAdam algorithm to enhance the optimization algorithm's performance by analyzing the impacts of changes and momentum during training. Wei Yuan [3] later proposed the EAdam algorithm, which outperforms the Adam algorithm by adjusting the position of the constant ε. Additionally, Mingrui Liu et al. [4] proposed the Adam-plus algorithm, which utilizes an adaptive step size based on the first-order momentum estimation paradigm, resulting in reduced

parameter tuning. Ilya Loshchilov [5] proposed two algorithms: AdamW and AdamWR. AdamWR is a variant of the Adam algorithm that corrects the weight decay problem in the algorithm, while AdamW is a variant of Adam with thermal restart. Lei Guan [6] proposed the Adaplus algorithm, which combines the advantages of Nesterov momentum and exact step-size adjustment. Adaplus is based on AdamW, which combines the advantages of the NAdam [7] and AdaBelief [8] algorithms and does not introduce additional hyper-parameters. Juyoung Yun [9] proposed the StochGradAdam algorithm, which utilizes selective gradient considerations for more reliable convergence. Zhang et al. [10] combined the warmup and cosine annealing algorithms to enhance the performance of the Adam algorithm in deep learning. Meanwhile, researchers have made improvements in the area of differential privacy. Qiaoyue Tang et al. [11] proposed the DP-Adam algorithm, which adds DP (differential privacy) noise that does not affect the first moment but adds a constant bias to the second moment. Qiaoyue Tang et al. [12] proposed the DP-AdamBC algorithm, which removes the bias in the estimation of the second moment and restores the desired behavior of the Adam algorithm. Lu Xia et al. [13] proposed the AdamL algorithm, which considers the information of the loss function to obtain a better generalization performance during optimization. In addition to other algorithms that enhance traditional optimization methods by introducing new techniques or combining multiple optimization methods, Kavosh Asadi et al. [14] found that resetting the optimizer improves performance and stability during the training process. Sebastian Bieringer et al. [15] proposed the AdamM-CMC algorithm, which, unlike the traditional Adam algorithm, utilizes Markov chain Monte Carlosampling from the posterior distribution to provide a more reliable estimate of uncertainty. The Adan [16] algorithm utilizes a new Nesterov momentum estimation (NME) method to estimate the first- and second-order moments in an adaptive gradient algorithm, accelerating convergence. Yichuan Shao et al. [17] proposed a lightweight, multi-scale neural network for detecting steel surface defects, addressing the shortcomings of the Adam algorithm in terms of convergence speed and generalization ability. Later, they developed a novel method for detecting dust on PV panel surfaces based on Pytorch [18]. Deep learning techniques have been increasingly utilized in healthcare. Gupta et al. [19] employed these techniques to segment brain tumors. Lisa Y.W. Tang [20] utilized a convolutional network to classify the severity of hairy glass-like cloudy opacity. Bhoj Raj Pandit [21] employed a convolutional neural network pooling layer multi-space image and the Adam algorithm for optimization to enhance the overall accuracy and predict lung cancer. Loris Nanni [22] proposed two DGrad-based deep networks to optimize the new Adam algorithm and introduced a scale factor in the learning rate to test the dataset for medical image classification.

This study focused on providing insight into the challenges faced by the Adam algorithm in terms of convergence. We found that while the Adam algorithm performs well in many cases, its performance may suffer in specific situations, such as when the dataset is highly correlated or when the learning rate decay is not appropriately set. By delving into these challenges, we aimed to reveal the limitations of the Adam algorithm and propose improvements to enhance its performance and stability in various situations. This study proposes a new optimization algorithm called Cycle-Norm-Adam (CN-Adam). The CEDLR algorithm aims to periodically adjust the learning rate to jump out of local optima and accelerate convergence. By periodically adjusting the learning rate during training, the CEDLR algorithm helps the optimization algorithm to explore more extensively in the search space, thus making it more likely to find the global optimum or a better solution. This periodic adjustment of the learning rate makes the algorithm better able to adapt to different data distributions and model structures, thus improving the efficiency and performance of the optimization algorithm. In addition, a gradient paradigm constraint is used to limit the size of the normalized gradient. Its main function is to prevent the problem of exploding or vanishing gradients, thus improving the stability of the optimization algorithm. By setting the threshold of the gradient paradigm number, the gradient paradigm constraint ensures that the size of the gradient is always kept within a reasonable range, avoiding the problems

that may be caused by a gradient that is too large or too small, such as unstable training or failure to converge. Combining both techniques, the CN-Adam algorithm performs well in deep learning and provides a feasible optimization strategy for solving practical problems.

## 2. CN-Adam Algorithm Design

### *2.1. Adam Optimization Algorithm*

The Adam optimizer is an optimization algorithm that combines the momentum method and the adaptive learning rate property [23] to train neural networks and optimize objective functions. The core idea is to dynamically adjust the learning rate based on the gradient profile of each parameter, as well as to accelerate convergence using the momentum term. The Adam algorithm optimizes the parameters by maintaining first-order and second-order moment estimates for each parameter. When updating the parameters, Adam combines the two estimates and performs bias corrections to ensure that the initial stage estimates are unbiased. The Adam optimizer also employs a technique known as exponential moving average to compute first-order moment estimates and second-order moment estimates of the gradient. By using exponential moving average, Adam is able to efficiently aggregate and update historical gradient information to better reflect the optimization direction of the parameters. In addition, the Adam algorithm is adaptive in that it can automatically adjust the size of the learning rate to the specifics of each parameter, which is particularly useful when dealing with parameters with different gradient distributions.

In each iteration, the gradient of the loss function with respect to the parameters is calculated using Equation (1).

$$g_t = \nabla_\theta f_t(\theta_t) \tag{1}$$

The parameter vector $\theta_t$ is updated each time. $g_t$ represents the loss function of the neural network in the $t$-th iteration, while $g_t = \nabla_\theta f_t(\theta_t)$ represents the gradient of the loss function of the neural network with respect to the parameters in the $t$-th iteration. The first-order and second-order moment estimates in the Adam algorithm represent the moving average of the first-order and second-order of the gradient. These estimates are calculated as shown in Equations (2) and (3).

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t \tag{2}$$

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2 \tag{3}$$

The variable $\beta_1$ is used to control the contribution of past gradients to the current estimate. Specifically, $\beta_2$ is used to control the contribution of squared past gradients to the current estimate.

The Adam optimization algorithm introduces a bias correction mechanism to address potential bias in the first-order moment estimates and second-order moment estimations when the decay rate is very small in the initial stage. The computation of bias correction is shown in Equations (4) and (5).

$$\hat{m}_t = m_t / (1 - \beta_1^t) \tag{4}$$

$$\hat{v}_t = v_t / (1 - \beta_2^t) \tag{5}$$

The update of parameter $\theta$ after each iteration is computed according to Equation (6).

$$\theta_{t+1} = \theta_t - \frac{lr}{\sqrt{v_t} + \varepsilon} \cdot \hat{m}_t \tag{6}$$

where $lr$ is the learning rate, and $\varepsilon$ is a small positive number to avoid a zero denominator.

One of the drawbacks of the Adam optimizer is its sensitivity to the learning rate. Although it is able to adaptively adjust the learning rate for each parameter, sometimes this adaptivity may cause the learning rate to decay too quickly during training, thus

affecting the convergence performance. In addition, since the Adam algorithm performs second-order moment estimations of the gradient, this makes it potentially underperform when dealing with non-smooth objective functions, especially when the gradient varies over a wide range or the objective function is highly non-convex. Therefore, when applying the Adam optimizer, attention needs to be paid to its sensitivity to the learning rate and gradient, as well as the instability and degradation of the convergence performance that may occur in specific scenarios.

### *2.2. Cyclic Exponential Decay Learning Rate*

Cyclical learning rate (CLR) is a method for optimizing the training process of neural networks, originally proposed by Leslie N. Smith [24]. The algorithm introduces the concept of periodically adjusting the learning rate. Unlike traditional learning rate scheduling methods, cyclical learning rates allow the learning rate to fluctuate periodically during the training process, rather than remaining fixed or decaying linearly. This periodically adjusted learning rate strategy adds greater flexibility and dynamics to the model training. During training, the cyclical learning rate cause the learning rate to periodically fluctuate within a predefined range, thus allowing the model to use different learning rates at different training stages. This flexibility allows the model to better adapt to different data distributions and complexities, thus improving the generalization ability of the model. In addition, cyclical learning rate can help the model jump out of the local optimal point and find the global optimal solution faster. Although cyclical learning rate needs to adjust some hyperparameters to define the period and range of the learning rate, they have proventheir effectiveness and superiority in many practical applications.

Triangular mode is the most commonly used mode in cyclical learning rate algorithms, where the learning rate periodically fluctuates over a fixed range, forming a triangular waveform. In this mode, the learning rate is first increased to a maximum value and then gradually decreased to a minimum value before repeating the process. The calculation of the triangular pattern with respect to the learning rate is shown in Equation (7).

$$lr\_delta = lr\_base + (lr\_max - lr\_base) \cdot \max(0, 1 - |\frac{iteration}{step\_size} - 2 \cdot cycle + 1|) \quad (7)$$

where $lr\_delta$ is the current learning rate. $lr\_base$ is the minimum value of the learning rate during the change process. $lr\_max$ is the maximum value of the learning rate during the change process. $step\_size$ is half the cycle length. $iteration$ is the number of parameter updates.

The cyclic exponential decay learning rate (CEDLR) method used in this paper was different from the traditional cyclical learning rate calculation method to calculate the learning rate. In this study, the calculation of the algorithm was modified, and the formulas are shown in Equations (8)–(11).

$$cycle = math.floor(1 + state['step'] / (2 * step\_size)) \quad (8)$$

$$x = |state['step'] - 2 \cdot cycle + 1| \quad (9)$$

$$lr = lr\_base + (lr\_max - lr\_base) \cdot \max(0, 1 - |\frac{state['step']}{step\_size} - 2 \cdot cycle + 1|) \quad (10)$$

$$lr\_delta = lr \cdot gamma^{state['step']} \quad (11)$$

where $lr\_delta$ is the current learning rate, $lr\_base$ is the minimum value of the learning rate during the change process, and $gamma$ is the decay coefficient. $lr\_max$ is the maximum value of the learning rate during the change process, $cycle$ indicates the current cycle of the loop, $state['step']$ is the total number of parameter updates throughout the training process of the model, and $step\_size$ is half the cycle length.

By comparing Formulas (8)–(11), it is evident that the calculation methods for iterations and state ['step'] are completely different. The calculation method in the original paper

updates parameters in each epoch by dividing the total number of samples by the batch size and rounding up. This means that multiple parameter update operations will be performed in each epoch, and each update is considered a new iteration. In this paper, the state ['step'] can determine the current training process based on its value and then adjust the learning rate as needed. The important advantage of using state ['step'] as the global step count is that it is an internal property of the optimizer, directly associated with the step count of parameter updates. This makes it more suitable as a benchmark for learning rate schedulers, as it ensures that the adjustment of learning rates is consistent with parameter updates.

By periodically adjusting the learning rate, the model can be explored and exploited at different learning rate levels during the training process, which helps to avoid falling into local optimal solutions. The exponential decay allows the learning rate to gradually decrease in the later stages of training, which helps to improve the convergence and generalization ability of the training. Taken together, this learning rate calculation method is able to provide appropriate learning rates for the model at different stages, which speeds up the training process and improves the model's performance.

### *2.3. Gradient Norm Constraint Strategy*

The gradient norm constraint strategy limits the size of the overall gradient vector by setting a threshold for the maximum number of gradient norms, thereby controlling the magnitude of parameter changes during each update. This constraint helps to prevent the excessive adjustment of model parameters and improve the stability and generalization ability of the model during training. It plays a significant role in the entire algorithm.

One of the advantages of a gradient paradigm constraint is that it prevents gradient explosion and gradient vanishing. In deep neural networks, due to the large number of network layers and the choice of activation function, the gradient may grow or decay exponentially in the back-propagation process, which will affect the stability and convergence of the model. The size of the gradient can be effectively controlled by the gradient paradigm constraint, thus stabilizing the training process of the model. The gradient paradigm constraint calculation formula is defined as shown in Equation (12).

$$grad = \begin{cases} grad \cdot \frac{grad\_norm\_constraint}{||grad||}, if ||grad|| > grad\_norm\_constraint \\ grad, otherwise \end{cases} \tag{12}$$

where $grad$ denotes the original gradient vector, $||grad||$ denotes the gradient vector's paradigm, and $grad\_norm\_constraint$ denotes the threshold of the gradient paradigm.

Equation (12) states that before performing the gradient update, the magnitude of the gradient vector is calculated. If the magnitude exceeds a set threshold, the vector is scaled to match the threshold. Otherwise, the original gradient vector remains unchanged.

Gradient norm constraints also help improve a model's ability to generalize. Excessive gradients can lead to the over-fitting of the model to the training data, and the gradient norm constraint defined in this article limits the size of model parameter updates. When combined with the learning rate modified by state ['step'], the experimental results demonstrate an accuracy that is 3.61% and 6.4% higher than that of the Adam algorithm, particularly in the CIFAR10 and Medical datasets.

### *2.4. CN-Adam Algorithm*

The Adam optimizer may lead to unstable training or slower convergence at higher learning rates. This is mainly because the learning rate adjustment mechanism of the Adam optimizer cannot adapt well to the needs of different learning rate parameters. Its learning rate adjustment mechanism has limited flexibility. To solve this problem, an improved CN-Adam algorithm is proposed with the following procedure.

Initialization: The initial learning rate $lr$, the minimum learning rate $lr\_base$, and the maximum learning rate $lr\_max$ are set;the first-order momentum and the second-order

momentum are initialized as zero vectors; and the number of steps per complete cycle *step_size* is set to 1400.

Cyclic exponential decay learning rate phase: The learning rate gradually increases to a maximum value for a specified number of steps and then gradually decreases. The current number of training steps and the cycle size determine the current cycle position, which is adjusted according to the position and the pre-set minimum and maximum learning rates according to Equations (8)–(11).

The gradient paradigm constraint phase: The first step is to check whether the gradient exceeds the set threshold. If it does, it may lead to unstable training. To stabilize the training, the gradient is constrained, usually by a scaling operation to ensure that it does not exceed the threshold. The scaling operation is performed by multiplying a scaling factor that controls the gradient's size, improving the training stability and convergence speed.

The first-order momentum and second-order momentum of the bias is computed for parameter updates. The Adam optimization algorithm's update rules are used, along with bias correction terms. To prevent gradient explosion, gradient paradigm constraints are applied. The parameter values are updated using the updated gradient and bias correction term. Once the parameters are updated, the current gradient is saved as the last gradient to be used in the next iteration.

The cyclic exponential decay learning rate algorithm can automatically adjust the learning rate to promote faster convergence and better generalization of the model. The cyclic exponential decay learning rate algorithm has high adaptability and dynamically adjusts the learning rate based on the performance of the model during training, which helps to avoid gradient explosion or vanishing problems. Combined with the proposed gradient paradigm constraints, it can help control the size of gradients and prevent the occurrence of gradient explosion problems. By combining the two, the optimizer can be optimized to appear more stable, as shown in Algorithm 1.

**Algorithm 1** CN-Adam

**Input**: initial point $x_0$, first moment decay $\beta_1$, second moment decay $\beta_2$, regularization constant $\varepsilon$
**Initialize** $m_0$ and $v_0 = 0$, $lr$,$lr\_base$,$lr\_max$,$step\_size$,$gamma$,$grad\_norm\_constraint$
**For** 0 to $step\_size$ do
$g_t = \nabla_\theta f_t(\theta_t)$
$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t$
$v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2$
$\hat{m}_t = m_t / (1-\beta_1^t)$
$\hat{v}_t = v_t / (1-\beta_2^t)$
$cycle = math.floor(1 + state['step'] / (2 * step\_size))$
$x = |state['step'] - 2 \cdot cycle + 1|$
$lr = lr\_base + (lr\_max - lr\_base) \cdot \max(0, 1 - |\frac{state['step']}{step\_size} - 2 \cdot cycle + 1|)$
$lr\_delta = lr \cdot gamma^{state['step']}$
**While** $lr\_delta = lr \cdot gamma^{state['step']}$ do
**If** $lr > lr\_max$ **or** $lr < lr\_base$
    **end while**
**If** $||grad|| > grad\_norm\_constraint$
$grad \leftarrow grad \cdot \frac{grad\_norm\_constraint}{||grad||}$
    $x_t = x_{t-1} - lr\hat{m}_t / (\sqrt{\hat{v}_t} + \varepsilon)$
**end for**
**Return** $x_t$

## 3. Experimental Design and Analysis of Results

### *3.1. Configuration of the Experimental Environment*

The main software versions applied in the experiment are shown in Table 1. This experiment applied the deep learning framework Pytorch Lightning, and the algorithm used was the improved CN-Adam algorithm based on the Adam algorithm, which was

jointly improved by the cyclic exponential decay learning rate algorithm and the customized gradient paradigm constraint method.

**Table 1.** Main software versions.

| Software | Version |
|---|---|
| Python | 3.10 |
| CUDA | cu118 |
| torch | 2.0.1 |
| torchvision | 0.15.0 |
| Lightning | 2.1.2 |
| wandb | 0.16.0 |

The entire experimental code was written using the PyTorch lightning framework. The programming language was Python, Python language version is 3.10, the torch version was 2.0.1, the torchvision version was 0.15.0, the CUDA version was cu118, the Lightning version was 2.1.2, and the wandb version was 0.16.0.

The CN-Adam algorithm's performance was tested using two commonly used datasets, MNIST and CIFAR10, as well as medical domain image classification experiments. The experiment included three datasets: MNIST, CIFAR10, and a medical dataset. MNIST is a grayscale image dataset of handwritten numbers with an image size of 28*28. CIFAR10 is a color image dataset containing different types of items with an image size of 32*32. The medical dataset is divided into two parts: the upper and lower gastrointestinal tracts. For this experiment, we used the dataset for the upper gastrointestinal system. The experiment involved eight classifications, primarily consisting of various grades of hemorrhoids, polyps, and ulcerative colitis. The image size was processed as 224*224. The experimental dataset is presented in Table 2.

**Table 2.** Experimental dataset.

| Dataset | Sample Size | Training Set | Validation Set | Test Set | Categories | Features |
|---|---|---|---|---|---|---|
| MNIST | 70,000 | 55,000 | 5000 | 10,000 | 10 | Grayscale images, few classifications, and difficult to recognize |
| CIFAR10 | 60,000 | 45,000 | 5000 | 10,000 | 10 | Color RGB images, fewer classifications, and difficult to recognize |
| Medical | 1885 | 900 | 485 | 500 | 8 | Color RGB images, fewer classifications, and difficult to recognize |

### *3.2. Experimental Results and Analysis*

The CN-Adam algorithm was extended and improved based on the Adam algorithm. It now includes the cyclic exponential decay learning rate algorithm and a gradient paradigm constraint strategy, making it more comprehensive and flexible. These improvements enhance the algorithm's generalization ability and convergence speed for the model. To evaluate the advantages of the CN-Adam algorithm over other optimization algorithms, we conducted experimental comparisons using various optimization algorithms, including SGD, AdaGrad, Adadelta, Adam, and two variants of Adam:the NAdam and StochGradAdam algorithms. Several experiments were conducted to compare the accuracy and loss values in the test set. The results proving the best performance were selected and bolded as shown in Table 3 to demonstrate the superiority of the CN-Adam algorithm. The table compares the experimental results of the different optimization algorithms.

**Table 3.** Comparison of experimental results of different optimization algorithms.

| Dataset | Optimization Algorithm | Accuracy | Loss |
|---|---|---|---|
| MNIST | SGD | 98.42% | 0.081 |
| | AdaGrad | 98.51% | 0.057 |
| | Adadelta | 96.44% | 0.13 |
| | Adam | 98.53% | 0.056 |
| | NAdam | 98.48% | 0.061 |
| | StochGradAdam | 98.09% | 0.07 |
| | CN-Adam | **98.54%** | 0.06 |
| CIFAR10 | SGD | 49.68% | 1.434 |
| | AdaGrad | 30.21% | 1.935 |
| | Adadelta | 23.95% | 2.032 |
| | Adam | 68.49% | 1.21 |
| | NAdam | 68.67% | 1.638 |
| | StochGradAdam | 68.07% | 1.04 |
| | CN-Adam | **72.10%** | **0.902** |
| Medical | SGD | 63.60% | 1.185 |
| | AdaGrad | 67.40% | 1.012 |
| | Adadelta | 56.00% | 2.261 |
| | Adam | 72.40% | 0.872 |
| | NAdam | 72.80% | 0.857 |
| | StochGradAdam | 70.20% | 1.055 |
| | CN-Adam | **78.80%** | **0.7245** |

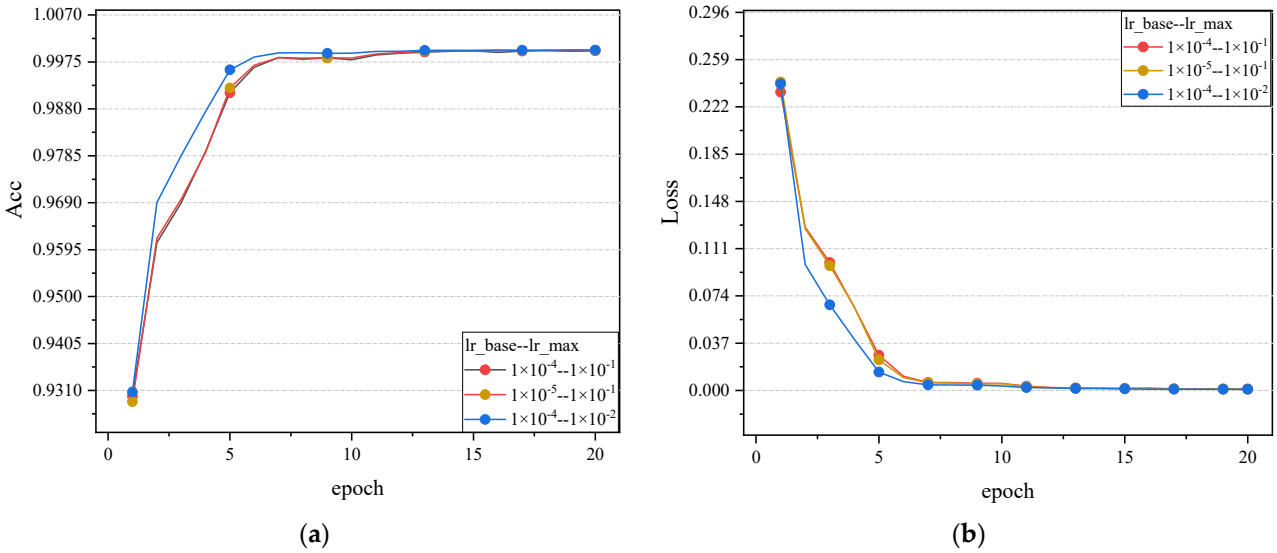
Experimental Setup

(1) Application domain: The algorithm used three datasets in this study, namely, the MNIST dataset on handwritten digit recognition, the CIFAR10 dataset of color images with 10 classifications, and the medical dataset in healthcare. The download paths of the corresponding datasets can be found in the author's GitHub code and the data availability statement of this article.
(2) Optimization algorithms: The experiments covered seven optimization algorithms, namely, SGD, AdaGrad, Adadelta, Adam, NAdam, StochGradAdam, and CN-Adam, aiming at comparing the performance differences between them.
(3) Batch size: The batch size used in each experiment was 128 to ensure the consistency and fairness of the experiment.
(4) Learning rate setting: The initial learning rate for all three datasets was 0.001. For the CN-Adam algorithm, the maximum learning rate was 0.01, and the minimum learning rate was 0.0001.
(5) Epoch size: To accurately assess the performance of each optimizer, all experiments were conducted with 100 epochs to ensure adequate and accurate model training.
(6) Adjustment of key parameters: Key parameters in the algorithm were fine-tuned based on different datasets to ensure the comparability and accuracy of the experimental results.
(7) Data preprocessing: Before conducting the experiments, necessary data processing operations, such as normalization, standardization, and data augmentation, were performed to ensure the quality and consistency of the input data.
(8) Experimental results: Several comparison experiments were conducted on the MNIST, CIFAR10, and medical datasets, taking into account factors such as Acc, loss, and GPU power consumption to fully demonstrate the advantages of the CN-Adam algorithm.

The CN-Adam algorithm aims to quickly discover locally optimal solutions to enhance the performance of the optimization algorithm by achieving optimal levels of test accuracy and loss values. To evaluate the performance of the CN-Adam algorithm in various neural networks and compare the results of the experiments, we extensively tested it on multiple datasets and neural network models. For the MNIST dataset, we chose a simple

fully connected neural network, and for the CIFAR10 and medical datasets, we chose a lightweight neural network, MobileNetV2.

The experiment involved comparing the performance of the CN-Adam algorithm in the MNIST dataset using three different learning rate size ranges and three different sets of hyperparameters. The results of this comparison are presented, focusing on the different combinations of learning rates. The first step of the experiment was to determine the range interval of the learning rate. This allowed for a better determination of the values of the other hyperparameters. Figure 1 shows the performance comparison for the MNIST dataset at different learning rate ranges.



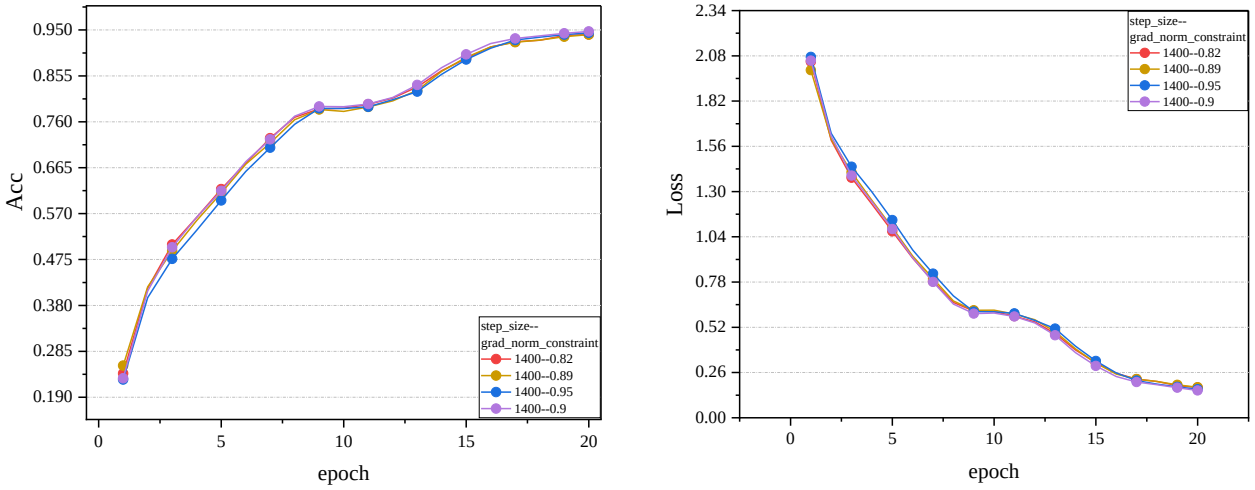

(**a**) (**b**)

**Figure 1.** Comparison of learning rates in the MNIST dataset: (**a**) comparison of accuracy; (**b**) comparison of loss values.

After analyzing Figure 1, it was determined that the combination of a minimum learning rate of $1 \times 10^{-4}$ and a maximum learning rate of $1 \times 10^{-2}$ was optimal. This combination converged quickly and maintained a stable performance. The early performance was poorer when the minimum learning rate was $1 \times 10^{-5}$, which may have been due to the learning rate being too small to adequately perform parameter updates. For the combination of a minimum learning rate of $1 \times 10^{-1}$, fast convergence could not be achieved early on. This may have been due to the learning rate being too large, resulting in unstable fluctuations in the model during training. Therefore, the optimal learning rate combination was between $1 \times 10^{-4}$ and $1 \times 10^{-2}$.

After determining the optimal learning rate, appropriate hyperparameter settings can improve the model stability and highlight the CN-Adam algorithm's effect. For example, the effects of the gradient paradigm constraints of different hyperparameters on the performance in CIFAR10 are shown in Figure 2. These results verify the effectiveness of the CN-Adam algorithm and provide guidance for its practical application.

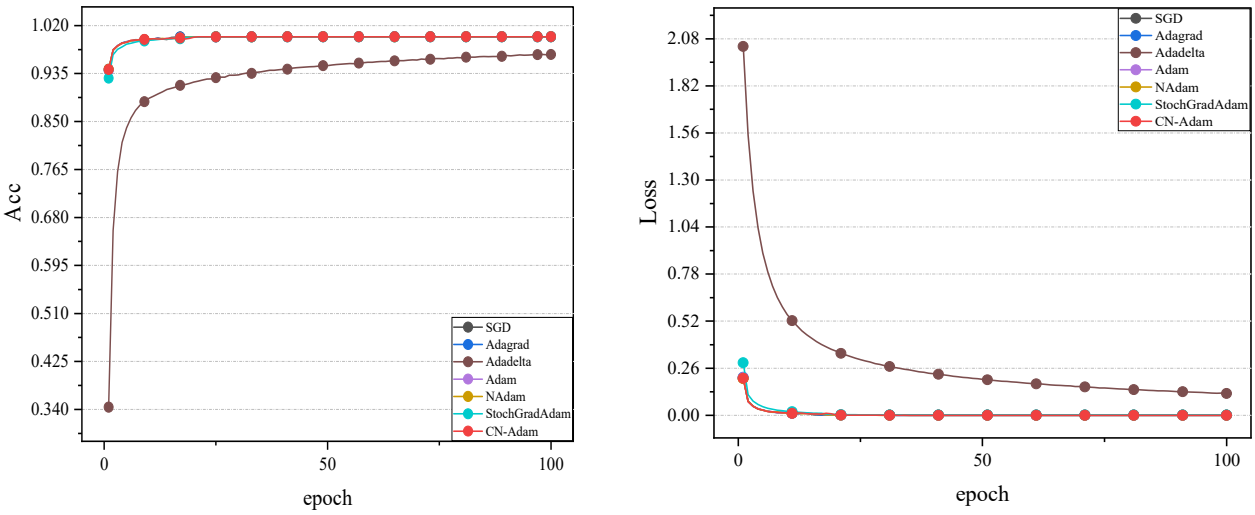
The Figure 2 above shows that adjusting the gradient paradigm constraint value led to significant performance differences, even with the same number of steps in the loop learning rate. The best accuracy and lowest loss values were achieved when the gradient paradigm constraint fetch value was set to 0.9. This setting resulted in an average accuracy improvement of approximately 1% compared with the other values tested. The significance of selecting an appropriate value for the gradient paradigm constraint to optimize the algorithm's convergence speed is emphasized. Additionally, the impact on the model performance is highlighted.




(**a**) (**b**)

**Figure 2.** Comparison of different key parameter values in the CIFAR10 dataset: (**a**) comparison of accuracy; (**b**) comparison of loss values.

Figure 3 shows a performance comparison of the seven optimization algorithms in the MNIST dataset.



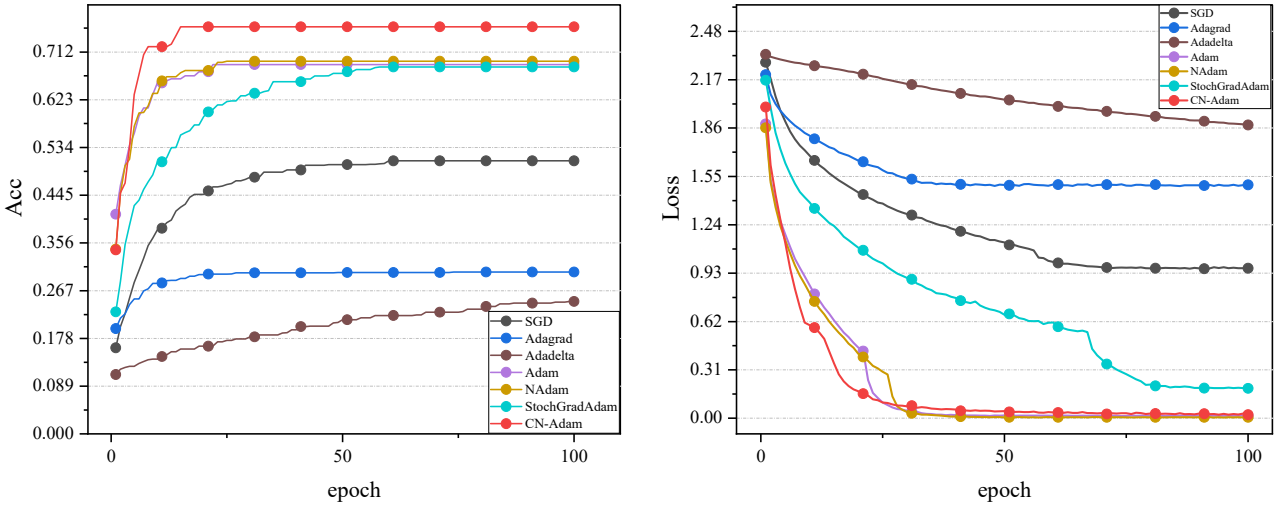

(**a**) (**b**)

**Figure 3.** Comparison of performance of seven algorithms in the MNIST dataset: (**a**) comparison of accuracy; (**b**) comparison of loss values.

The performance comparison of the seven optimization algorithms for the CIFAR10 dataset is shown in Figure 4.

As shown in Figure 4, the CN-Adam algorithm could quickly reach the optimal solution in this dataset, and the accuracy was improved by 3.61%, 3.43%, and 4.03%, and the loss value was reduced by 0.308, 0.736, and 0.138 compared with the accuracy of the Adam algorithm, the NAdam algorithm, and the StochGradAdam algorithm, which are based on the Adam algorithm and improved by the Adam algorithm. The fast convergence of the CN-Adam algorithm not only improves the training efficiency of the model but also enables the model to reach the desired performance level faster. Among them, the periodic learning rate adjustment strategy of the cyclic exponential decay learning rate algorithm brings a broader search capability to the optimization algorithm, which helps it to jump out of the local optimal solution and accelerate the convergence, thus further improving

the model performance. Meanwhile, the introduction of the gradient paradigm constraint technique effectively controls the size of the gradient, avoids the exploding or vanishing gradient problem, and enhances the stability of the optimization algorithm.




(**a**) (**b**)

**Figure 4.** Comparison of performance of seven algorithms in the CIFAR10 dataset: (**a**) comparison of accuracy; (**b**) comparison of loss values.

Due to the specificity of the medical dataset, a cross-validation strategy was used in this dataset to demonstrate the experimental results, and the validation and test sets were used to compare the effectiveness of the optimization algorithms in terms of accuracy and loss values. Figure 5 shows a comparison of the accuracy of the seven optimization algorithms in the validation set, and the accuracies of the SGD and Adadelta algorithms are significantly lower than those of the other optimization algorithms. Although the accuracy of the CN-Adam algorithm was slightly lower than that of the Adam algorithm and NAdam algorithm in the early stage of training, the accuracy of the CN-Adam algorithm significantly increased in the late stage of training as the number of training rounds increased, and it achieved good results in the test set. The accuracy was higher than that of the Adam algorithm by 6.4%, the NAdam algorithm by 6%, and StochGradAdam algorithm by 8.6%. This observation reveals the superior performance of the CN-Adam algorithm in dealing with complex data such as medical datasets. The problems with the Adam optimization algorithm have been addressed to some extent. Although there may be performance degradation in the initial phase, the performance gradually improves as the algorithm understands the data more deeply and learns the process. This dynamic learning capability allows the CN-Adam algorithm to continuously adjust its strategy and gradually optimize the model parameters, ultimately achieving results that outperform other optimization algorithms in the later training stages. Its robustness and adaptability enable it to better cope with the challenges in the data domain, providing strong support for the modeling and analysis of complex data.

The loss values of the seven optimization algorithms in the test set are compared in Figure 6. The results show that the loss value of the Adadelta algorithm was significantly higher than that of the other six optimization algorithms. The loss value of the CN-Adam algorithm was significantly lower than those of the other six algorithms, which were reduced by 0.1475, 0.1325, and 0.3305 compared with the Adam algorithm, the NAdam algorithm, which is based on the Adam algorithm, the improved NAdam algorithm, and the StochGradAdam algorithm. Combined with the validation set and test set results, these results show that the optimization algorithm demonstrates a superior performance in the medical domain. This highlights the importance of optimization algorithms when

applied in specific domains and emphasizes the need to select an appropriate optimization algorithm for optimal performance in the medical domain.

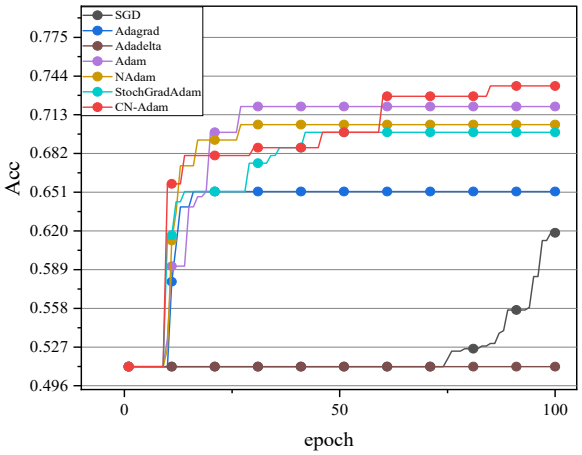



**Figure 5.** Comparison of validation accuracies of seven optimization algorithms in medical dataset.

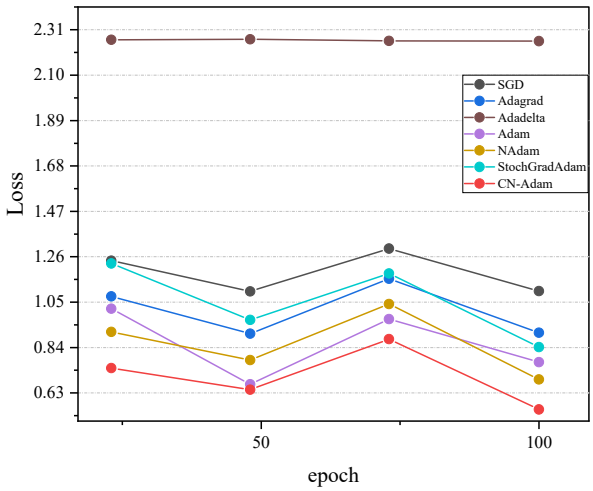



**Figure 6.** Comparison of test loss values of seven optimization algorithms in medical dataset.

Figure 7 compares the GPU power wattage for the CIFAR10 dataset, showing that the CN-Adam algorithm consumed less power than the other algorithms. However, its training time was slightly longer due to the dynamic adjustment of the learning rate. Despite this, the CN-Adam algorithm demonstrated its strength in resource utilization by maintaining the lowest level of power consumption. This emphasizes the significance of taking into account both the power consumption and training time when choosing an optimization algorithm to attain optimal performance and efficiency in real-world applications.

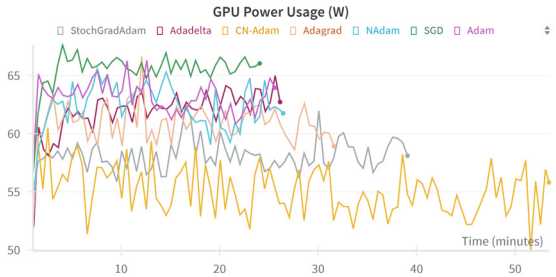



**Figure 7.** Comparison of GPU power wattage for the CIFAR10 dataset.

Figure 8 displays the GPU power consumption percentages for the medical dataset. It is evident that the CN-Adam algorithm quickly recognized the medical dataset and had a shorter training time than all the other algorithms. This indicates its strong performance in the medical domain, emphasizing its efficiency in processing medical data. Selecting the appropriate optimization algorithm in the medical field is essential for achieving the prompt recognition and efficient processing of medical data.

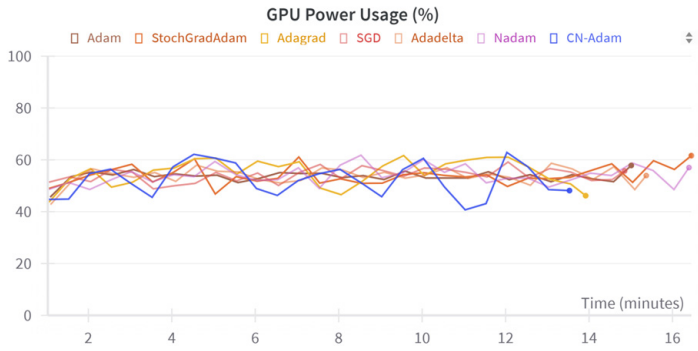



**Figure 8.** Percentages of GPU power consumption for medical dataset.

## 4. Conclusions

This study proposed a new optimization algorithm called CN-Adam, which aims to address the shortcomings of the Adam optimization algorithm. The experimental results demonstrate that the CN-Adam algorithm outperformed other methods, including the StochGradAdam algorithm, in multiple datasets. The CN-Adam algorithm enhances the stability and convergence speed of the Adam model while achieving more precise tuning by combining the cyclic exponential decay learning rate algorithm with gradientparadigm constraints. Although it may require more computational resources than other optimization algorithms, the CN-Adam algorithm improves the model performance and is applicable to areas such as medical image processing. This approach improves the model performance and generalization ability, providing new momentum to the development of deep learning.

**Author Contributions:** Conceptualization, methodology, and writing—original draft preparation, J.Y.; software, project administration, and resources, Y.S.; data curation, W.Z.; writing—review and editing, supervision, and formal analysis, H.S.; funding acquisition, Q.Z., L.X. and L.Z. All authors have read and agreed to the published version of the manuscript.

**Funding:** 1. The Liaoning Provincial Department of Education Basic Research Project for Higher Education Institutions (General Project), Shenyang University of Technology, Research on Optimization Design of Wind Turbine Cone Angle Based on Fluid Physics Method (LJKZ0159); 2. the Basic Research Project of the Liaoning Provincial Department of Education "Training and Application of Multimodal Deep Neural Network Models for Vertical Fields" (project number: JYTMS20231160); 3. Research on the Construction of a New Artificial Intelligence Technology and High-Quality Education Service Supply System in the 14th Five Year Plan for Education Science in Liaoning Province, 2023–2025 (project number: JG22DB488); 4. the "Chunhui Plan" of the Ministry of Education, Research on an Optimization Model and Algorithm for Microgrid Energy Scheduling Based on Biological Behavior (project no. 202200209); and 5. the Shenyang Science and Technology Plan "Special Mission for Leech Breeding and Traditional Chinese Medicine Planting in Dengshibao Town, Faku Country" (project no. 22-319-2-26).

**Data Availability Statement:** The Python code used in this paper is located at https://github.com/icecream1024/CN.git (accessed on 29 March 2024). The website for the standard CIFAR10 dataset is https://www.kaggle.com/datasets/gazu468/cifar10-classification-image (accessed on 1 May 2022). The website for the medical datasets is https://doi.org/10.1038/s41597-020-00622-y (accessed on 28 August 2020).

**Conflicts of Interest:** The authors declare no conflicts of interest.

## References


1. Jiang, Y.; Liu, J.; Xu, D.; Mandic, D.P. UAdam: Unified Adam-Type Algorithmic Framework for Non-Convex Stochastic Optimization. *arXiv* **2023**. [CrossRef]
2. Liu, L.; Jiang, H.; He, P.; Chen, W.; Liu, X.; Gao, J.; Han, J. On the Variance of the Adaptive Learning Rate and Beyond. *arXiv* **2021**. [CrossRef]
3. Yuan, W.; Gao, K.-X. EAdam Optimizer: How ε Impact Adam. *arXiv* **2020**. [CrossRef]
4. Liu, M.; Zhang, W.; Orabona, F.; Yang, T. Adam$^{+}$: A Stochastic Method with Adaptive Variance Reduction. *arXiv* **2020**. [CrossRef]
5. Loshchilov, I.; Hutter, F. Decoupled Weight Decay Regularization. *arXiv* **2017**. [CrossRef]
6. Guan, L. AdaPlus: Integrating Nesterov Momentum and Precise Stepsize Adjustment on Adamw Basis. In Proceedings of the ICASSP 2024—2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Seoul, Republic of Korea, 14–19 April 2024; pp. 5210–5214. [CrossRef]
7. Dozat, T. Incorporating Nesterov Momentum into Adam. February 2016. Available online: https://openreview.net/forum?id=OM0jvwB8jIp57ZJjtNEZ (accessed on 19 February 2024).
8. Zhuang, J.; Tang, T.; Ding, Y.; Tatikonda, S.C.; Dvornek, N.; Papademetris, X.; Duncan, J. AdaBelief Optimizer: Adapting Stepsizes by the Belief in Observed Gradients. In *Advances in Neural Information Processing Systems*; Neural Information Processing Systems Foundation, Inc. (NeurIPS): La Jolla, CA, USA, 2020; Volume 33, pp. 18795–18806.
9. Yun, J. StochGradAdam: Accelerating Neural Networks Training with Stochastic Gradient Sampling. *arXiv* **2023**. [CrossRef]
10. Zhang, C.; Shao, Y.; Sun, H.; Xing, L.; Zhao, Q.; Zhang, L. The WuC-Adam algorithm based on joint improvement of Warmup and cosine annealing algorithms. *Math. Biosci. Eng.* **2023**, *21*, 1270–1285. [CrossRef]
11. Tang, Q.; Lécuyer, M. DP-Adam: Correcting DP Bias in Adam's Second Moment Estimation. *arXiv* **2023**. [CrossRef]
12. Tang, Q.; Shpilevskiy, F.; Lécuyer, M. DP-AdamBC: Your DP-Adam Is Actually DP-SGD (Unless You Apply Bias Correction). *arXiv* **2023**. [CrossRef]
13. Xia, L.; Massei, S. AdamL: A fast adaptive gradient method incorporating loss function. *arXiv* **2023**. [CrossRef]
14. Asadi, K.; Fakoor, R.; Sabach, S. Resetting the Optimizer in Deep RL: An Empirical Study. In *Advances in Neural Information Processing Systems*; Neural Information Processing Systems Foundation, Inc. (NeurIPS): La Jolla, CA, USA, 2023; Volume 36, pp. 72284–72324.
15. Bieringer, S.; Kasieczka, G.; Steffen, M.F.; Trabs, M. AdamMCMC: Combining Metropolis Adjusted Langevin with Momentum-based Optimization. *arXiv* **2023**. [CrossRef]
16. Xie, X.; Zhou, P.; Li, H.; Lin, Z.; Yan, S. Adan: Adaptive Nesterov Momentum Algorithm for Faster Optimizing Deep Models. *arXiv* **2023**. [CrossRef]
17. Shao, Y.; Fan, S.; Sun, H.; Tan, Z.; Cai, Y.; Zhang, C.; Zhang, L. Multi-Scale Lightweight Neural Network for Steel Surface Defect Detection. *Coatings* **2023**, *13*, 1202. [CrossRef]
18. Shao, Y.; Zhang, C.; Xing, L.; Sun, H.; Zhao, Q.; Zhang, L. A new dust detection method for photovoltaic panel surface based on Pytorch and its economic benefit analysis. *Energy AI* **2024**, *16*, 100349. [CrossRef]
19. Gupta, A.; Dixit, M.; Mishra, V.K.; Singh, A.; Dayal, A. Brain Tumor Segmentation from MRI Images Using Deep Learning Techniques. In *Advanced Computing*; Springer Nature: Cham, Switzerland, 2023; pp. 434–448. [CrossRef]
20. Tang, L.Y.W. Severity classification of ground-glass opacity via 2-D convolutional neural network and lung CT scans: A 3-day exploration. *arXiv* **2023**. [CrossRef]
21. Pandit, B.R.; Alsadoon, A.; Prasad, P.W.; Al Aloussi, S.; Rashid, T.A.; Alsadoon, O.H.; Jerew, O.D. Deep Learning Neural Network for Lung Cancer Classification: Enhanced Optimization Function. *Multimed. Tools Appl.* **2023**, *82*, 6605–6624. [CrossRef]
22. Nanni, L.; Manfe, A.; Maguolo, G.; Lumini, A.; Brahnam, S. High performing ensemble of convolutional neural networks for insect pest image detection. *Ecol. Inform.* **2022**, *67*, 101515. [CrossRef]
23. Kingma, D.P.; Ba, J. Adam: A Method for Stochastic Optimization. *arXiv* **2017**. [CrossRef]
24. Smith, L.N. Cyclical Learning Rates for Training Neural Networks. In Proceedings of the 2017 IEEE Winter Conference on Applications of Computer Vision (WACV), Santa Rosa, CA, USA, 24–31 March 2017; pp. 464–472. [CrossRef]