

# AdaNorm: Adaptive Gradient Norm Correction based Optimizer for CNNs

Shiv Ram Dubey\*, Satish Kumar Singh\*, Bidyut Baran Chaudhuri†

\*Computer Vision and Biometrics Lab, Indian Institute of Information Technology, Allahabad

†Techno India University, Kolkata, India and Indian Statistical Institute, Kolkata, India

srdubey@iiita.ac.in, sk.singh@iiita.ac.in, bidyutbaranchaudhuri@gmail.com

## Abstract

The stochastic gradient descent (SGD) optimizers are generally used to train the convolutional neural networks (CNNs). In recent years, several adaptive momentum based SGD optimizers have been introduced, such as Adam, diffGrad, Radam and AdaBelief. However, the existing SGD optimizers do not exploit the gradient norm of past iterations and lead to poor convergence and performance. In this paper, we propose a novel AdaNorm based SGD optimizers by correcting the norm of gradient in each iteration based on the adaptive training history of gradient norm. By doing so, the proposed optimizers are able to maintain high and representative gradient throughout the training and solves the low and atypical gradient problems. The proposed concept is generic and can be used with any existing SGD optimizer. We show the efficacy of the proposed AdaNorm with four state-of-the-art optimizers, including Adam, diffGrad, Radam and AdaBelief. We depict the performance improvement due to the proposed optimizers using three CNN models, including VGG16, ResNet18 and ResNet50, on three benchmark object recognition datasets, including CIFAR10, CIFAR100 and TinyImageNet. Code: <https://github.com/shivram1987/AdaNorm>.

## 1. Introduction

In recent years, Convolutional Neural Networks (CNNs) have become the major parametric model to solve the Computer Vision problems [8], such as Object Recognition [28], [11], Object Localization [24], [23], Image Segmentation [9], Face Recognition [27], [2], Image Retrieval [4], Biomedical Image Analysis [29], and many more. The training of CNN models is performed to learn the parameters of the network on the training set of data.

In practice, the batch-wise Stochastic Gradient Descent (SGD) based optimization techniques are used to train the CNN models. The parameters/weights are first initialized using some approach, such as random initialization, Xavier initialization [7], He initialization [10], etc. Then, the pa-

rameters are updated by utilizing the gradient of objective function w.r.t. the corresponding parameter in multiple iterations [26]. The vanilla SGD approach [1] updates the parameters in the opposite direction of gradient by a small step-size, called as learning rate. However, it suffers with various challenges, such as zero gradient at local minimum and saddle regions, severe variations in gradient in different directions, same step-size used for each parameter update irrespective of its behaviour, and bad gradient due to batch-wise computation. The SGD with Momentum (i.e., SGDM) [30] tackles the first two issues by considering the exponential moving average (EMA) of gradient for parameter update. The EMA of gradient builds the velocity in the direction of consistent gradient for faster convergence. The step-size problem is addressed by AdaGrad [6] which divides the step-size by the root of sum of past squared gradient. However, it leads to dying learning rate problem in the later stage of training, which is fixed in RMSProp [13] by dividing the step-size with root of the EMA of squared gradient. The Adam optimizer [16] combines the concept of SGDM and RMSProp and proposes adaptive moments. The first and second moments are computed as EMA of gradients and squared gradients, respectively. Adam uses first moment to update the parameters and second moment to control the step-size. Adam optimizer has been used successfully with various CNN models for different computer vision problems. In order to deal with the effect of bad batch-wise gradient on the effective learning rate the AMSGrad [22] uses maximum of past squared gradients to control the learning rate, rather than exponential average. However, the AMSGrad does not deal with bad gradient used for parameter updates, which is taken care in the proposed AdaNorm optimizers.

The Adam optimizer suffers near the minimum due to high moment leading to overshooting of minimum and oscillation near minimum [5], [20], [32]. Recent optimizers have tried to tackle this issue, such as diffGrad [5] introduces a friction coefficient based on the local gradient behaviour to reduce the learning rate near minimum; Rectified Adam (i.e., Radam) [20] rectifies the variance of the

---

**Algorithm 1: Adam Optimizer**

---

**Initialize:**  $\theta_0, \mathbf{m}_0 \leftarrow 0, \mathbf{v}_0 \leftarrow 0, t \leftarrow 0$   
**Hyperparameters:**  $\alpha, \beta_1, \beta_2$   
**While**  $\theta_t$  not converged  
   $t \leftarrow t + 1$   
   $\mathbf{g}_t \leftarrow \nabla_{\theta} f_t(\theta_{t-1})$   
   $\mathbf{m}_t \leftarrow \beta_1 \mathbf{m}_{t-1} + (1 - \beta_1) \mathbf{g}_t$   
   $\mathbf{v}_t \leftarrow \beta_2 \mathbf{v}_{t-1} + (1 - \beta_2) \mathbf{g}_t^2$   
  **Bias Correction**  
     $\hat{\mathbf{m}}_t \leftarrow \mathbf{m}_t / (1 - \beta_1^t), \hat{\mathbf{v}}_t \leftarrow \mathbf{v}_t / (1 - \beta_2^t)$   
  **Update**  
     $\theta_t \leftarrow \theta_{t-1} - \alpha \hat{\mathbf{m}}_t / (\sqrt{\hat{\mathbf{v}}_t} + \epsilon)$

---

adaptive learning rate and converts Adam into SGDM based on the variance threshold; and AdaBelief [32] considers the EMA of square of difference between the gradient and first order moment (i.e., belief information) to control the learning rate. The other variants of Adam includes Nostalgic Adam (NosAdam) [15] which gives more weight to the past gradients to incorporate the long-term memory. However, NosAdam miss to rectify the norm of the gradients. The AdaBound [21] performs clipping to make the optimizer more robust to extreme learning rates, caused by adaptive momentum. The AdaBound approach can be seen as the post-correction of learning rates. Similarly, the adaptive and momental upper bounds are used in AdaMod [3] to avoid the large learning rates in the initial iterations of Adam. The AdamP [12] has shown that the decay in learning rate might lead to sub-optimal solution and can be tackled by getting rid of the radial component. The Yogi [31] utilizes the limit of variance in the stochastic gradients to control the learning rate. The AngularGrad [25] utilizes the change in gradient orientation to control the learning rate. In order to control the learning rate, decay based SGD approaches have been also exploited [14] [17]. Though the existing optimization methods try to control the learning rate by exploiting different properties of gradients, they still suffer due to inconsistent gradients. In this paper, we tackle this issue through the gradient norm correction to make it historically consistent throughout the training iterations.

In this paper we tackle the above mentioned issues with the help of gradient norm correction by exploiting the history of gradient norm. The contributions are as follows:

1. We propose an AdaNorm approach by exploiting the EMA of gradient norm of past iterations. The proposed AdaNorm rectifies the gradient norm based on the training history to better maintain the consistent and informative gradient.
2. The proposed AdaNorm approach is generic and can be used with any existing adaptive SGD optimizer. We use the proposed AdaNorm with Adam [16], diffGrad

---

**Algorithm 2: AdamNorm Optimizer**

---

**Initialize:**  $\theta_0, \mathbf{m}_0 \leftarrow 0, \mathbf{v}_0 \leftarrow 0, e_0 \leftarrow 0, t \leftarrow 0$   
**Hyperparameters:**  $\alpha, \beta_1, \beta_2, \gamma$   
**While**  $\theta_t$  not converged  
   $t \leftarrow t + 1$   
   $\mathbf{g}_t \leftarrow \nabla_{\theta} f_t(\theta_{t-1})$   
   $\mathbf{g}_{norm} \leftarrow L_2 \text{Norm}(\mathbf{g}_t)$   
   $e_t = \gamma e_{t-1} + (1 - \gamma) \mathbf{g}_{norm}$   
   $\mathbf{s}_t = \mathbf{g}_t$   
  **If**  $e_t > \mathbf{g}_{norm}$   
     $\mathbf{s}_t = (e_t / \mathbf{g}_{norm}) \mathbf{g}_t$   
   $\mathbf{m}_t \leftarrow \beta_1 \mathbf{m}_{t-1} + (1 - \beta_1) \mathbf{s}_t$   
   $\mathbf{v}_t \leftarrow \beta_2 \mathbf{v}_{t-1} + (1 - \beta_2) \mathbf{g}_t^2$   
  **Bias Correction**  
     $\hat{\mathbf{m}}_t \leftarrow \mathbf{m}_t / (1 - \beta_1^t), \hat{\mathbf{v}}_t \leftarrow \mathbf{v}_t / (1 - \beta_2^t)$   
  **Update**  
     $\theta_t \leftarrow \theta_{t-1} - \alpha \hat{\mathbf{m}}_t / (\sqrt{\hat{\mathbf{v}}_t} + \epsilon)$

---

[5], Radam [20] and AdaBelief [32] optimizers and propose AdamNorm, diffGradNorm, RadamNorm and AdaBeliefNorm optimizers, respectively.

3. We include an intuitive explanation and convergence proof for the proposed optimizer. We also show the impact of the proposed AdaNorm approach on the behaviour of gradient norm experimentally.
4. We perform a rigorous experimental study on three benchmark datasets, including CIFAR10, CIFAR100 and TinyImageNet for object recognition to demonstrate the efficacy of the proposed AdaNorm based optimizers. The impacts of hypermeter, AdaNorm on second moment, learning rate and batch size are also studied in the experiments.

We organize this paper by presenting the proposed AdaNorm optimizers in Section 2, Intuitive Explanation and Convergence Analysis in Section 3, Experimental settings in Section 4, Results & discussion in Section 5, Ablation study in Section 6 and Conclusion in Section 7.

## 2. Proposed AdaNorm Optimizer

Let consider a network ( $f$ ) represented by its parameters  $\theta$  to be trained using SGD approach in an iterative manner. The parameters are initialized before start of the training and represented as  $\theta_0$ . In any given  $t^{th}$  iteration, the gradient of objective function w.r.t. the parameters (i.e.,  $\mathbf{g}_t$ ) is computed using chain-rule and expressed as,

$$\mathbf{g}_t = \nabla_{\theta} f_t(\theta_{t-1}) \quad (1)$$

where  $f_t$  is the model at  $t^{th}$  iteration,  $\theta_{t-1}$  represent the parameters in the previous iteration, and  $\nabla_{\theta}$  represent the gradient over parameters  $\theta$ .

The existing optimizers, such as Adam (see Algorithm 1), diffGrad, Radam, and AdaBelief use the  $\mathbf{g}_t$  to compute the Exponential Moving Average (EMA) of gradients ( $\mathbf{m}_t$ ) which is used to update the parameters. However, the gradient  $\mathbf{g}_t$  is computed as an average on a batch of randomly drawn training samples. Hence, it might not be representative, not consistent with the past gradient behaviour and prone to be bad. In this paper we tackle this problem by correcting the norm of the gradient of current batch with the help of historical gradient norm.

Let  $g_{norm} = \|\mathbf{g}_t\|_2$  is the L2-Norm of the current gradient vector  $\mathbf{g}_t = (g_{t,1}, g_{t,2}, \dots, g_{t,k})$ . The computation of the  $g_{norm}$  can be given as,

$$g_{norm} = \sqrt{\sum_{i=1}^k (g_{t,i})^2} \quad (2)$$

where  $g_{t,i}$  is the  $i^{th}$  element of  $\mathbf{g}_t$  and  $k$  is the number of elements in  $\mathbf{g}_t$ .

Let represent the norm corrected gradient as  $\mathbf{s}_t$ . The computation of  $\mathbf{s}_t$  is proposed as,

$$\mathbf{s}_t = \begin{cases} (e_t/g_{norm})\mathbf{g}_t, & \text{if } e_t > g_{norm} \\ \mathbf{g}_t, & \text{otherwise} \end{cases} \quad (3)$$

where  $e_t$  is the historical gradient norm computed in the  $t^{th}$  iteration using the norm of past gradients, i.e., previous iterations. We use the EMA approach to compute  $e_t$  as,

$$e_t = \gamma e_{t-1} + (1 - \gamma)g_{norm} \quad (4)$$

where  $g_{norm}$  is the L2-Norm of the current gradient  $\mathbf{g}_t$ ,  $e_{t-1}$  is the historical gradient norm computed in the previous  $(t-1)^{th}$  training iteration with  $e_0 = 0$  as the initial value before the start of the training and  $\gamma$  is a hyperparameter to control the contribution of past historical gradient norm and current gradient norm in the computation of new historical gradient norm. The impact of  $\gamma$  is analyzed in the experiments section. The proposed gradient norm correction step makes the norm of the current gradient to be at least the historical gradient norm. Inherently, it forces the current gradient to be better aligned and consistent with the behaviour of the previous gradients and tackles the problem of bad gradients in existing methods. Moreover, it reduces the dependency on batch size and makes the training of the deep network more effective and stable.

We integrate the proposed AdaNorm concept of gradient norm correction with Adam [16] and propose AdamNorm optimizer. We use the gradient with corrected norm ( $\mathbf{s}_t$ ) to compute the first moment  $\mathbf{m}_t$  in the proposed AdamNorm optimizer, given as,

$$\mathbf{m}_t = \beta_1 \mathbf{m}_{t-1} + (1 - \beta_1)\mathbf{s}_t \quad (5)$$

where  $\mathbf{m}_{t-1}$  and  $\mathbf{m}_t$  are the first moment in  $(t-1)^{th}$  and  $t^{th}$  iterations, respectively,  $\mathbf{m}_0$  is initialized with 0, and  $\beta_1$  is a hyperparameter. However, we use the original gradient ( $\mathbf{g}_t$ ) to compute the second moment, given as,

$$\mathbf{v}_t = \beta_2 \mathbf{v}_{t-1} + (1 - \beta_2)\mathbf{g}_t^2 \quad (6)$$

where  $\mathbf{v}_{t-1}$  and  $\mathbf{v}_t$  are the second moment in  $(t-1)^{th}$  and  $t^{th}$  iterations, respectively,  $\mathbf{v}_0$  is initialized with 0, and  $\beta_2$  is a hyperparameter. As the second moment is used to control the learning rate, we avoid to use the norm corrected gradient in its computation as it may lead to significantly lower effective step-size and hamper the learning capability. The impact of gradient norm correction on second moment is analyzed in the experiments.

By following the Adam [16], we perform the bias correction of moments to avoid very high step-size in the initial training iterations as follows,

$$\widehat{\mathbf{m}}_t \leftarrow \mathbf{m}_t / (1 - \beta_1^t) \quad (7)$$

$$\widehat{\mathbf{v}}_t \leftarrow \mathbf{v}_t / (1 - \beta_2^t) \quad (8)$$

where  $t$  is the current iteration number,  $\widehat{\mathbf{m}}_t$  and  $\widehat{\mathbf{v}}_t$  are the first and second moment after bias correction, respectively.

Finally, the parameters of the network is updated based on  $\widehat{\mathbf{m}}_t$  and  $\widehat{\mathbf{v}}_t$  as follows,

$$\boldsymbol{\theta}_t \leftarrow \boldsymbol{\theta}_{t-1} - \alpha \widehat{\mathbf{m}}_t / (\sqrt{\widehat{\mathbf{v}}_t} + \epsilon) \quad (9)$$

where  $\boldsymbol{\theta}_{t-1}$  is the parameter after  $(t-1)^{th}$  training iteration,  $\boldsymbol{\theta}_t$  is the parameter after the current training iteration, and  $\alpha$  is the learning rate used to compute the effective step-size for the parameter update. The steps of the proposed AdamNorm optimizer is summarized in Algorithm 2 with highlighted changes in Blue color w.r.t. the Adam.

Note that the proposed gradient norm correction using historical gradient norm is a generic idea and can be integrated with any existing SGD optimization technique. We just described above the steps of AdamNorm, i.e., the integration of the proposed concept with Adam [16]. However, in order to show the generalization of the gradient norm correction approach, we also integrate it with the recent state-of-the-art optimizers, including diffGrad [5], Radam [20] and AdaBelief [32] optimizers and propose diffGradNorm, RadamNorm and AdaBeliefNorm optimizers, respectively. The Algorithms of diffGrad, diffGradNorm, Radam, RadamNorm, AdaBelief and AdaBeliefNorm are provided in Supplementary.

### 3. Intuitive Explanation and Convergence Analysis

#### 3.1. Intuitive Explanation

In order to justify the importance of the gradient norm correction, we provide an intuitive explanation through Fig.

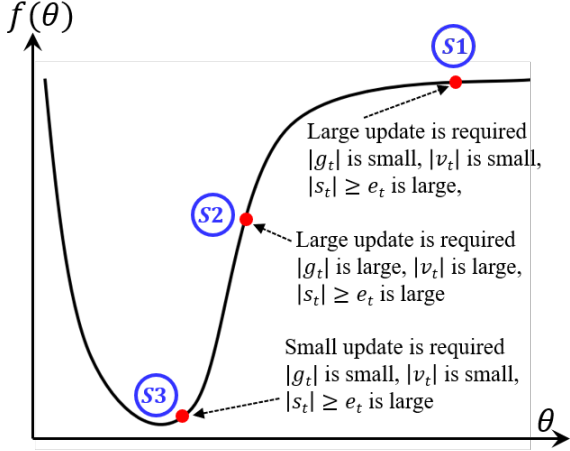


Figure 1. Typical scenarios depicting the importance of adaptive parameter update in optimization [32].

1 that how the proposed AdaNorm approach provides a promising trade-off between large and small weight updates with the help of three typical scenarios in optimization on a one dimensional optimization curvature (i.e., S1, S2 and S3). The bias correction step is ignored for simplicity. The gradient norm  $g_{norm}$  is considered as  $|g_t|$  for one dimensional optimization.

The scenario **S1** depicts the flat region which is very common in optimization. In such region, an ideal optimizer expects the large update, but the gradient  $g_t$  is very small. The small  $g_t$  limits the  $m_t$  in case of Adam leading to still small update. However, the  $m_t$  is large in case of AdamNorm leading to large update as  $s_t$  is large due to  $|s_t| \geq e_t$  which is maintained to be sufficiently large historically over the training epochs. The  $v_t$  is small and equally helpful in both the cases.

The scenario **S2** depicts the large gradient region in the optimization landscape. In such region, an ideal optimizer expects the large update, which is supported by large  $g_t$ . The  $m_t$  in Adam is large in this case leading to large update. However, the  $m_t$  in AdamNorm is at least the  $m_t$  in Adam leading to large update. It shows that AdamNorm can perform at least as good as Adam in large gradient region. The effect of  $v_t$  is similar in both the cases.

The scenario **S3** depicts the steep and narrow valley region in the optimization landscape, which mimics the minimum of function. In such region, an ideal optimizer expects the small update, which is supported by small  $g_t$ . The Adam leads to small  $m_t$  leading to small update, but the AdamNorm leads to relatively large  $m_t$  leading to relatively large update which might be favourable to unwanted local minimum. In case of minimum, the  $m_t$  in AdamNorm will become small in few iterations of parameter updates near minimum which will eventually lead to convergence. The  $v_t$  behaves equally bad in both the cases.

### 3.2. Convergence Analysis

We use the online learning framework proposed in [33] to show the convergence property of AdamNorm similar to Adam [16]. Assume  $f_1(\theta), f_2(\theta), \dots, f_T(\theta)$  as the convex cost functions in an unknown sequence. We compute the regret bound as follows,

$$R(T) = \sum_{t=1}^T [f_t(\theta_t) - f_t(\theta^*)] \quad (10)$$

where  $f_t(\theta_t)$  is the  $t^{th}$  online guess,  $f_t(\theta^*)$  is the best parameter setting from a feasible set  $\chi$  with  $\theta^* = \arg \min_{\theta \in \chi} \sum_{t=1}^T f_t(\theta)$ . It is observed that the regret bound of AdamNorm is similar to Adam, i.e.,  $O(\sqrt{T})$ . We provide the convergence proof of the AdamNorm in Supplementary. Let  $g_{t,i}$  and  $s_{t,i}$  are the gradient and the norm rectified gradient, respectively, in the  $t^{th}$  iteration for the  $i^{th}$  element,  $g_{1:t,i} = [g_{1,i}, g_{2,i}, \dots, g_{t,i}] \in \mathbb{R}^t$  and  $s_{1:t,i} = [s_{1,i}, s_{2,i}, \dots, s_{t,i}] \in \mathbb{R}^t$  are the gradient vector and the norm rectified gradient vector, respectively, for the  $i^{th}$  parameter over all iterations up to  $t$ , and  $\eta \triangleq \frac{\beta_1^2}{\sqrt{\beta_2}}$ .

**Theorem 1.** *Let the gradients for function  $f_t$  are bounded (i.e.,  $\|g_{t,\theta}\|_2 \leq G$  and  $\|g_{t,\theta}\|_\infty \leq G_\infty$  for all  $\theta \in \mathbb{R}^d$ ). Let the distance produced by AdamNorm between any  $\theta_t$  are also bounded (i.e.,  $\|\theta_n - \theta_m\|_2 \leq D$  and  $\|\theta_n - \theta_m\|_\infty \leq D_\infty$  for any  $m, n \in \{1, \dots, T\}$ ). Let  $\eta \triangleq \frac{\beta_1^2}{\sqrt{\beta_2}}$ ,  $\beta_1, \beta_2 \in [0, 1)$  satisfy  $\frac{\beta_1^2}{\sqrt{\beta_2}} < 1$ ,  $\alpha_t = \frac{\alpha}{\sqrt{t}}$ , and  $\beta_{1,t} = \beta_1 \lambda^{t-1}$ ,  $\lambda \in (0, 1)$  where  $\lambda$  is typically very close to 1, e.g.,  $1 - 10^{-8}$ . For all  $T \geq 1$ , the AdamNorm shows the following guarantee:*

$$\begin{aligned} R(T) &\leq \frac{D^2}{2\alpha(1-\beta_1)} \sum_{i=1}^d \sqrt{T \hat{v}_{T,i}} \\ &\quad + \frac{\alpha(1+\beta_1)G_\infty^3}{(1-\beta_1)\sqrt{1-\beta_2}(1-\gamma)^2 G^2} \sum_{i=1}^d \|g_{1:T,i}\|_2 \\ &\quad + \sum_{i=1}^d \frac{D_\infty^2 G_\infty \sqrt{1-\beta_2}}{2\alpha(1-\beta_1)(1-\lambda)^2} \end{aligned}$$

Note that the additive term over the dimension ( $d$ ) can be much smaller than its upper bound  $\sum_{i=1}^d \|g_{1:T,i}\|_2 < dG_\infty \sqrt{T}$  and  $\sum_{i=1}^d \sqrt{T \hat{v}_{T,i}} < dG_\infty \sqrt{T}$ . In general,  $O(\log d \sqrt{T})$  is achieved by adaptive methods such as Adam and AdamNorm which is better than the  $O(\sqrt{dT})$  of non-adaptive methods. The following corollary depicts the convergence of average regret of AdamNorm.

**Corollary 1.** *Consider the bounded gradients for function  $f_t$  (i.e.,  $\|g_{t,\theta}\|_2 \leq G$  and  $\|g_{t,\theta}\|_\infty \leq G_\infty$ ) for all  $\theta \in \mathbb{R}^d$ . Also, assume that the AdamNorm produces the bounded distance between any  $\theta_t$  (i.e.,  $\|\theta_n - \theta_m\|_2 \leq D$*

Table 1. Classification results in terms of accuracy (%) on CIFAR10, CIFAR100 and TinyImageNet datasets using Adam, diffGrad, Radam and AdaBelief without and with the proposed AdaNorm technique. The value of  $\gamma$  is set to 0.95 in this experiment and results are computed as an average over three independent runs.

CNN Models	Classification accuracy (%) using different optimizers without and with AdaNorm							
	Adam		diffGrad		Radam		AdaBelief	
	Adam	AdamNorm	diffGrad	diffGradNorm	Radam	RadamNorm	AdaBelief	AdaBeliefNorm
Results on CIFAR10 Dataset								
VGG16	92.55	<b>92.83</b> ( $\uparrow$ 0.30)	92.76	<b>92.87</b> ( $\uparrow$ 0.12)	92.94	<b>93.14</b> ( $\uparrow$ 0.22)	92.71	<b>92.81</b> ( $\uparrow$ 0.11)
ResNet18	93.54	<b>93.78</b> ( $\uparrow$ 0.26)	93.49	<b>93.98</b> ( $\uparrow$ 0.52)	93.82	<b>93.89</b> ( $\uparrow$ 0.07)	93.63	<b>93.66</b> ( $\uparrow$ 0.03)
ResNet50	93.83	<b>94.01</b> ( $\uparrow$ 0.19)	93.81	<b>94.23</b> ( $\uparrow$ 0.45)	94.14	<b>94.21</b> ( $\uparrow$ 0.07)	94.1	<b>94.16</b> ( $\uparrow$ 0.06)
Results on CIFAR100 Dataset								
VGG16	67.29	<b>69.15</b> ( $\uparrow$ 2.76)	68.19	<b>68.31</b> ( $\uparrow$ 0.18)	70.69	<b>70.77</b> ( $\uparrow$ 0.11)	68.92	<b>69.24</b> ( $\uparrow$ 0.46)
ResNet18	71.09	<b>73.11</b> ( $\uparrow$ 2.84)	73.5	<b>73.64</b> ( $\uparrow$ 0.19)	73.22	<b>73.34</b> ( $\uparrow$ 0.16)	72.72	<b>73.31</b> ( $\uparrow$ 0.81)
ResNet50	71.88	<b>75.53</b> ( $\uparrow$ 5.08)	75.06	<b>75.49</b> ( $\uparrow$ 0.57)	74.95	<b>75.39</b> ( $\uparrow$ 0.59)	<b>75.53</b>	75.49 ( $\downarrow$ 0.05)
Results on TinyImageNet Dataset								
VGG16	41.93	<b>44.67</b> ( $\uparrow$ 6.53)	42.91	<b>43.49</b> ( $\uparrow$ 1.35)	43.84	<b>45.02</b> ( $\uparrow$ 2.69)	44.23	<b>44.79</b> ( $\uparrow$ 1.27)
ResNet18	47.73	<b>49.57</b> ( $\uparrow$ 3.86)	49.34	<b>49.80</b> ( $\uparrow$ 0.93)	48.73	<b>50.50</b> ( $\uparrow$ 3.63)	49.25	<b>49.99</b> ( $\uparrow$ 1.50)
ResNet50	48.98	<b>54.44</b> ( $\uparrow$ 11.15)	51.32	<b>53.75</b> ( $\uparrow$ 4.73)	51.63	<b>52.87</b> ( $\uparrow$ 2.40)	53.57	<b>54.44</b> ( $\uparrow$ 1.62)

and  $\|\theta_n - \theta_m\|_\infty \leq D_\infty$  for any  $m, n \in \{1, \dots, T\}$ ). For all  $T \geq 1$ , the proposed AdamNorm optimizer shows the following guarantee:

$$\frac{R(T)}{T} = O\left(\frac{1}{\sqrt{T}}\right).$$

Thus,  $\lim_{T \rightarrow \infty} \frac{R(T)}{T} = 0$ .

Theoretically, the convergence rate in terms of regret bounds for AdamNorm is similar to Adam-type optimizers (i.e.,  $O(\sqrt{T})$ ) [16], [5], [20], [32], which is computed in the worst possible case. However, the empirical analysis suggests that the AdamNorm outperforms Adam mainly because the cases as detailed in Section 3.1, which occur more frequently.

## 4. Experimental Settings

This section provides the details of CNN models used, datasets used and training settings.

**CNN Models Used:** In order to validate the efficacy of the proposed optimizers three CNN models, including VGG16 [28], ResNet18 and ResNet50 [11], are used in the experiments. The VGG16 is a simple CNN model, whereas the ResNet18 and ResNet50 are the residual connection based CNN models. The ResNet50 is a deep CNN model as compared to the VGG16 and ResNet18.

**Datasets Used:** We validate the performance of the proposed optimizers on three standard visual recognition datasets, including CIFAR10 [18], CIFAR100 [18], and TinyImageNet [19]. The CIFAR10 and CIFAR100 datasets contain 50000 images for training and 10000 images for testing. The CIFAR10 contains 10 object classes with equal number of samples. However, the CIFAR100 contains 100 object classes with equal number of samples. The

CIFAR100 is a fine-grained dataset. The TinyImageNet dataset contains 200 object classes with 500 training images per class (i.e., total 100000 training images) and 50 test images per class (i.e., total 10000 test images).

**Training Settings:** We perform the experiments using the Pytorch framework and train all the CNN models using Google Colab based freely available computational resources with single GPU. The training is performed for 100 Epochs with a batch size of 64. The learning rate is set to 0.001 initially and dropped to 0.0001 after 80 Epoch of training. For a fair comparison we consider the same common hyperparameters for all the optimizers, i.e.,  $\beta_1 = 0.9$  and  $\beta_2 = 0.999$ . The training and test images are normalized as per the standard practice. The data augmentations with random cropping, random horizontal flipping and normalization with mean (0.4914, 0.4822, 0.4465) and standard deviation (0.2023, 0.1994, 0.2010) are performed during training. Only normalization is used during testing.

## 5. Experimental Results and Discussion

This section provides the results comparison, experimental convergence analysis and the impact of the AdaNorm on the norm of the gradients.

### 5.1. Results Comparison

The results comparison of the proposed gradient norm correction based optimizers are presented in Table 1 in terms of the accuracy (%). We use four state-of-the-art adaptive optimizers (i.e., Adam [16], diffGrad [5], Radam [20] and AdaBelief [32]) for the results comparison by applying the proposed concept with these optimizers. The results are compared using VGG16, ResNet18 and ResNet50 models on CIFAR10, CIFAR100 and TinyImageNet datasets.

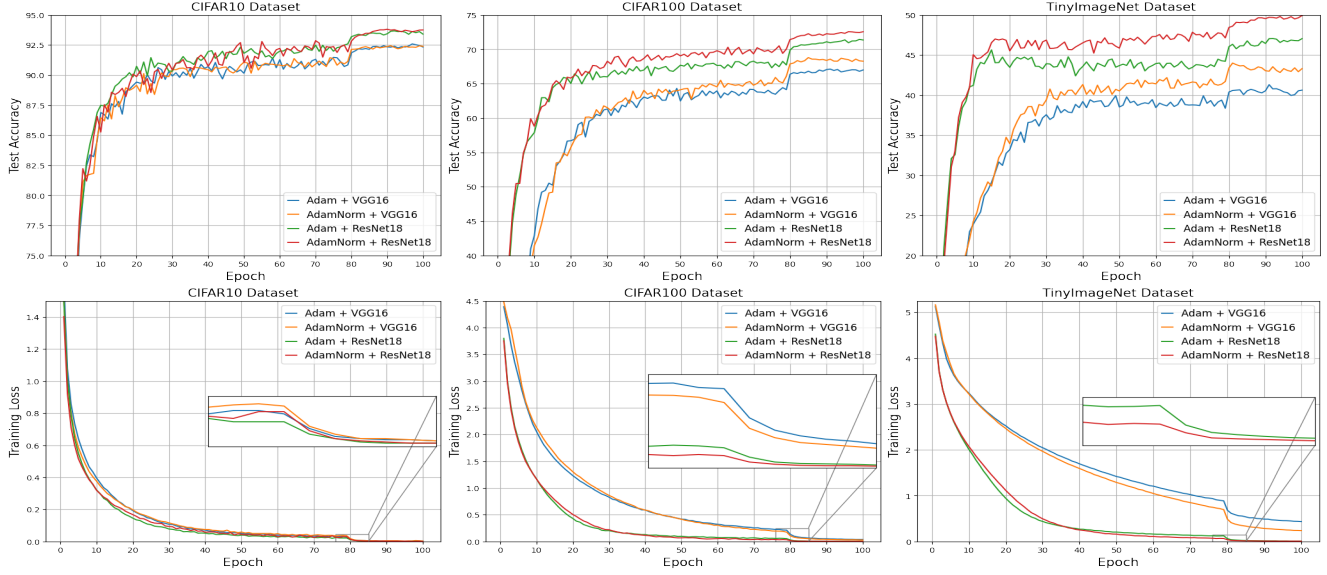


Figure 2. Test Accuracy (top row) and Training Loss (bottom row) vs Epoch plots using the Adam and AdamNorm optimizers for VGG16 and ResNet18 models on CIFAR10 (left), CIFAR100 (middle) and TinyImageNet (right) datasets. The value of  $\gamma$  is 0.95 in AdamNorm in this experiment. (Best viewed in color)

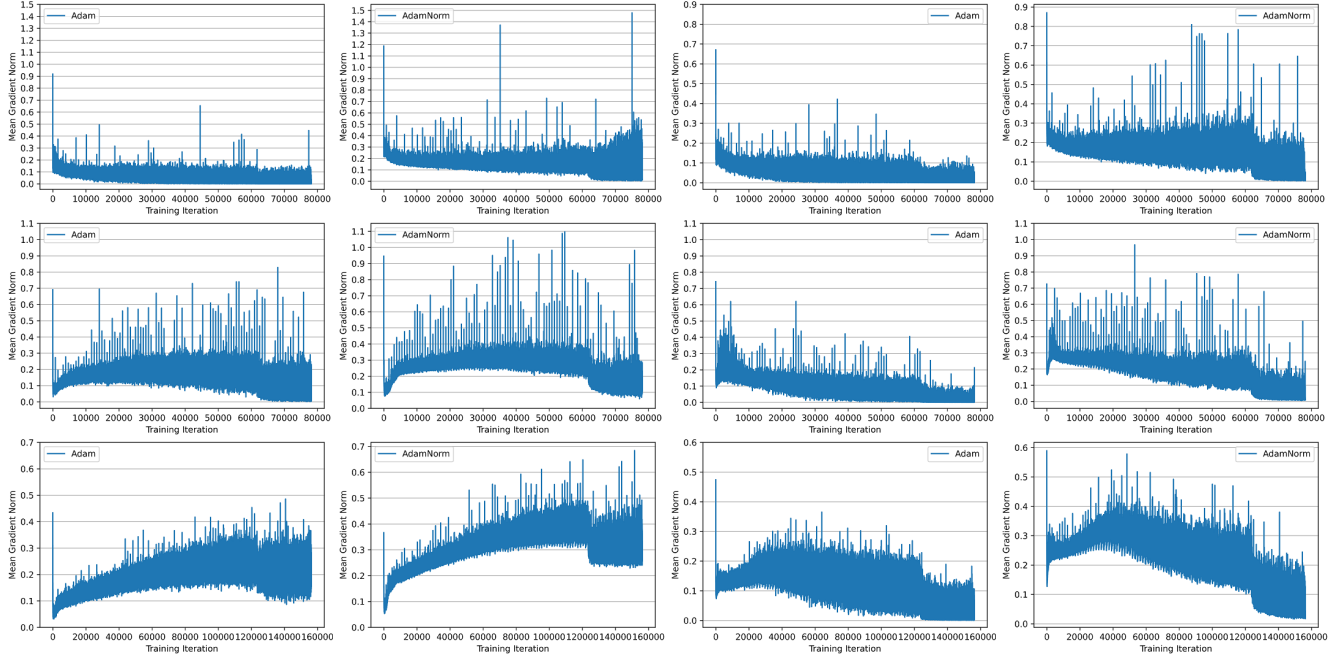


Figure 3. The Mean Gradient Norm using the Adam and AdamNorm optimizers (i.e., without and with the proposed history based gradient norm correction, respectively) at different training iterations. The plots in first, second and third row correspond to CIFAR10, CIFAR100 and TinyImageNet datasets, respectively. The plots in first and second column are computed using VGG16 and the plots in third and fourth column are computed using ResNet18. The gradient norm using the optimizers with proposed method is significantly greater than the corresponding vanilla optimizers. Note that the change in gradient norm at Epoch 80 is due to drop in the learning rate.

geNet datasets. The value of  $\gamma$  is set to 0.95 in this experiment and results are computed as an average over three independent runs. The higher results for an optimizer is highlighted in bold. The improvement in % due to the prop-

osed gradient norm correction is also highlighted in Table 1 with  $\uparrow$  symbol. It can be noticed that the accuracy due to the proposed optimizers is improved in almost all the cases. The performance is significantly improved on TinyIma-

Table 2. The effect of hyperparameter  $\gamma$  used in the EMA of the proposed history based gradient norm computation. The results are computed as an average over three independent runs using AdamNorm optimizer.

Hyperparameter ( $\gamma$ )	CIFAR10 Dataset			CIFAR100 Dataset			TinyImageNet Dataset		
	VGG16	ResNet18	ResNet50	VGG16	ResNet18	ResNet50	VGG16	ResNet18	ResNet50
0.9	92.56	93.72	93.95	68.85	73.06	<b>75.53</b>	44.85	<b>50.47</b>	<b>54.72</b>
0.95	92.83	<b>93.78</b>	94.01	69.15	73.11	<b>75.53</b>	44.67	49.57	54.44
0.99	<b>92.88</b>	93.55	<b>94.11</b>	<b>69.33</b>	<b>73.13</b>	75.43	<b>45.08</b>	49.41	54.54
0.999	92.66	93.68	93.54	69.18	73.16	75.48	44.71	50.45	54.14

Table 3. The results by applying the normalized gradient in different combination of first and second order moment. The value of  $\gamma$  is set to 0.95 in this experiment and results are computed as an average over three independent runs.

Normalized Gradient Setting	CIFAR10 Dataset			CIFAR100 Dataset			TinyImageNet Dataset		
	VGG16	ResNet18	ResNet50	VGG16	ResNet18	ResNet50	VGG16	ResNet18	ResNet50
1 <sup>st</sup> Moment	<b>92.83</b>	<b>93.78</b>	<b>94.01</b>	<b>69.15</b>	<b>73.11</b>	<b>75.53</b>	<b>44.67</b>	<b>49.57</b>	<b>54.44</b>
2 <sup>nd</sup> Moment	92.74	93.67	93.36	68.86	72.65	74.30	43.43	49.17	54.01
Both Moments	92.69	93.61	93.10	69.08	72.95	74.54	43.77	49.19	53.73

geNet dataset with highest improvement of 11.15% using AdamNorm optimizer as compared to Adam for ResNet50 model. The consistent improvement in the performance using different CNN models and optimizers confirm the importance of the gradient norm correction based on the history of the gradient norm.

## 5.2. Experimental Convergence Analysis

In order to highlight the improved convergence due to the proposed gradient norm correction based optimizer, we plot the test accuracy (top row) and training loss (bottom row) obtained at every epoch using Adam and AdamNorm (with  $\gamma = 0.95$ ) optimizers for VGG16 and ResNet18 models on CIFAR10, CIFAR100 and TinyImageNet datasets in Fig. 2. The test accuracy plots depict that the performance of the proposed AdamNorm is consistently better than the Adam on CIFAR100 and TinyImageNet datasets and slightly better on CIFAR10 dataset. The training loss curve for the AdamNorm is also better than the Adam on CIFAR100 and TinyImageNet dataset, while it is comparable on CIFAR10 dataset. From the training loss plots on CIFAR100 and TinyImageNet datasets, it is clear that the Adam optimizer initially converges faster, but get saturated soon due to the lack of consistent gradients over the training epochs. However, the proposed AdamNorm optimizer makes the consistent updates due to the norm corrected gradients used for updates and leads to significantly lower loss as compared to the Adam optimizer. It confirms the need of history based gradient norm correction for better optimization of CNNs.

## 5.3. Impact of Proposed Gradient Norm Correction

The proposed gradient norm correction aims to enforce the gradient norm at any training iteration to follow the trend of gradient norms of past training iterations. In order to observe the impact of the gradient norm correction, we plot the mean gradient norm of the Adam and AdamNorm

at each training iteration in Fig. 3 using VGG16 (1<sup>st</sup> and 2<sup>nd</sup> columns) and ResNet18 (3<sup>rd</sup> and 4<sup>th</sup> columns) models on CIFAR10, CIFAR100 and TinyImageNet datasets, in 1<sup>st</sup>, 2<sup>nd</sup> and 3<sup>rd</sup> row, respectively. It can be seen that the gradient norm of the AdamNorm is much higher and representative than the Adam in all the cases. It can also be observed that the gradient norm of the AdamNorm is better aligned with the historical trend set by the past training iterations. The improved representation of the gradient norm is the key to the performance improvement of the proposed AdaNorm based optimizers.

## 6. Ablation Study

This ablation study presents the effect of AdaNorm hyperparameter, second moment, learning rate & batch size.

### 6.1. Impact of AdaNorm Hyperparameter

In the proposed approach, the history of gradient norm is accumulated using EMA of gradient norms using a hyperparameter ( $\gamma$ ) in (4). In the results comparison we use the value of  $\gamma$  as 0.95. However, in this experiment, we compute the results using the proposed AdamNorm optimizer for  $\gamma = \{0.9, 0.95, 0.99, 0.999\}$  using the VGG16, ResNet18 and ResNet50 models on the CIFAR10, CIFAR100 and TinyImageNet datasets and report in Table 2. The results suggest that the value of  $\gamma$  is not recommended to be very high such as 0.999. Overall, relatively higher  $\gamma$  such as 0.99 is better suitable for the CIFAR10 and CIFAR100 datasets. However, relatively lower  $\gamma$  such as 0.90 leads to better performance on the TinyImageNet dataset. This behaviour is justified from the fact that the number of training iterations on the TinyImageNet is much higher as compared to the CIFAR10 and CIFAR100 datasets.



Table 4. The impact of learning rate ( $\alpha$ ). After 80 training Epochs,  $\alpha$  is divided by 10. The  $\gamma$  is set to 0.95 in this experiment and results are computed as an average over three independent runs.

Model	Adam Optimizer			AdamNorm Optimizer		
$\alpha$	0.01	0.001	0.0001	0.01	0.001	0.0001
CIFAR10 Dataset						
VGG16	92.21	<b>92.55</b>	92.02	92.30	<b>92.83</b>	92.27
ResNet18	92.90	<b>93.54</b>	93.07	93.18	<b>93.78</b>	93.13
CIFAR100 Dataset						
VGG16	65.93	67.29	<b>68.57</b>	66.19	<b>69.15</b>	68.75
ResNet18	66.84	71.09	<b>72.95</b>	68.42	73.11	<b>73.14</b>
TinyImageNet Dataset						
VGG16	36.78	41.93	<b>44.55</b>	41.41	44.67	<b>46.65</b>
ResNet18	45.50	47.73	<b>48.87</b>	46.94	49.57	<b>51.08</b>

## 6.2. Impact of AdaNorm on Second Moment

In the proposed approach, the gradient norm correction is only applied on the first moment. In this experiment, we compute the results by using the gradient norm correction in second moment also (i.e., using  $s_t^2$  instead of  $g_t^2$  in (6)). Basically, we compute the results using the proposed AdamNorm optimizer by applying the gradient norm correction in three settings, a) in 1<sup>st</sup> moment only, b) in 2<sup>nd</sup> moment only, and c) in both 1<sup>st</sup> and 2<sup>nd</sup> moments. The results are summarized for VGG16, ResNet18 and ResNet50 models on CIFAR10, CIFAR100 and TinyImageNet datasets in Table 3. It is evident that the performance of the proposed optimizer is best when the gradient norm correction is applied only on the 1<sup>st</sup> moment, because the 2<sup>nd</sup> moment controls the learning rate and applying the gradient norm correction on the 2<sup>nd</sup> moment hampers the effective step-size leading to poor performance.

## 6.3. Impact of Learning Rate

We also study the impact of learning rate ( $\alpha$ ) on the proposed AdamNorm optimizer. The classification accuracies are summarized in Table 4 on CIFAR10, CIFAR100 and TinyImageNet datasets using VGG16 and ResNet18 models for Adam and AdamNorm optimizers under different settings of learning rate, i.e.,  $\alpha = 0.01, 0.001, 0.0001$ . Note that the learning rate is divided by 10 after Epoch no. 80 in all the experiments. The results are reported as an average over three runs. The value of  $\gamma$  is set to 0.95. It is observed that the performance of AdamNorm is always better than Adam with same learning rate schedule. The results suggest that the smaller learning rate is better suitable on TinyImageNet datasets. However, the original considered learning rate (i.e., 0.001) is reasonable on CIFAR10 and CIFAR100 datasets using the AdamNorm optimizer.

Table 5. The impact of batch size ( $BS$ ). The  $\alpha$  is 0.001 the  $\gamma$  is set to 0.95 in this experiment. Results are computed as an average over three independent runs.

Model	Adam Optimizer			AdamNorm Optimizer		
BS	32	64	128	32	64	128
CIFAR10 Dataset						
VGG16	<b>92.56</b>	92.55	92.33	92.80	<b>92.83</b>	92.57
ResNet18	93.45	93.54	<b>93.60</b>	93.51	<b>93.78</b>	93.69
CIFAR100 Dataset						
VGG16	67.58	67.29	<b>67.80</b>	68.81	69.15	<b>69.45</b>
ResNet18	70.13	71.09	<b>71.62</b>	<b>73.43</b>	73.11	72.90
TinyImageNet Dataset						
VGG16	<b>42.11</b>	41.93	41.77	42.83	<b>44.67</b>	43.25
ResNet18	46.77	47.73	<b>48.31</b>	49.38	49.57	<b>50.59</b>

## 6.4. Impact of Batch Size

We also report the results by considering different batch sizes ( $BS$ ), such as 32, 64, and 128, in Table 5 for the Adam and AdamNorm optimizers on CIFAR10, CIFAR100 and TinyImageNet datasets using VGG16 and ResNet18 models. The results are computed as an average over three runs. The values of  $\alpha$  and  $\gamma$  are 0.001 and 0.95, respectively. It is observed that the performance of the proposed AdamNorm optimizer is always better than the performance of Adam for all the batch sizes. The results of Adam are mostly better with large batch size. However, the results of AdamNorm are better for all the batch sizes in some or other cases. It shows that the proposed optimizer is more robust to the batch size, because the norm of the gradients are corrected which reduces the dependency upon the batch size.

## 7. Conclusion

In this paper, we propose a gradient norm correction for the adaptive SGD optimizers based on the history of gradient norm. The proposed approach improves the representation of the gradient by boosting its norm to at least the historical gradient norm. The proposed approach is beneficial under flat and high gradient regions to improve the weight updates. The proposed approach is generic. We use it with Adam, diffGrad, Radam and AdaBelief optimizers and propose AdamNorm, diffGradNorm, RadamNorm and AdaBeliefNorm optimizers, respectively. The performance of the optimizers are significantly improved when used with the proposed gradient norm correction approach on CIFAR10, CIFAR100 and TinyImageNet datasets using VGG16, ResNet18 and ResNet50 models. The smaller value of hyperparameter  $\gamma$  is better suitable on TinyImageNet dataset. The proposed approach is suitable with only first moment. The smaller learning rate is preferred with the proposed AdamNorm on TinyImageNet dataset. The effect of batch size becomes negligible due to the proposed gradient norm correction.



## References

- [1] Léon Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of the COMPSTAT*, pages 177–186, 2010.
- [2] Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 4690–4699, 2019.
- [3] Jianbang Ding, Xuancheng Ren, Ruixuan Luo, and Xu Sun. An adaptive and momental bound method for stochastic learning. *arXiv preprint arXiv:1910.12249*, 2019.
- [4] Shiv Ram Dubey. A decade survey of content based image retrieval using deep learning. *IEEE Transactions on Circuits and Systems for Video Technology*, 2021.
- [5] Shiv Ram Dubey, Soumendu Chakraborty, Swalpa Kumar Roy, Snehasis Mukherjee, Satish Kumar Singh, and Bidyut Baran Chaudhuri. Diffgrad: an optimization method for convolutional neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 2019.
- [6] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul):2121–2159, 2011.
- [7] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 249–256, 2010.
- [8] Jiuxiang Gu, Zhenhua Wang, Jason Kuen, Lianyang Ma, Amir Shahroudy, Bing Shuai, Ting Liu, Xingxing Wang, Gang Wang, Jianfei Cai, et al. Recent advances in convolutional neural networks. *Pattern Recognition*, 77:354–377, 2018.
- [9] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *IEEE International Conference on Computer Vision*, pages 2961–2969, 2017.
- [10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *IEEE International Conference on Computer Vision*, pages 1026–1034, 2015.
- [11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.
- [12] Byeongho Heo, Sanghyuk Chun, Seong Joon Oh, Dongyoon Han, Sangdoo Yun, Youngjung Uh, and Jung-Woo Ha. Slowing down the weight norm increase in momentum-based optimizers. *arXiv preprint arXiv:2006.08217*, 2020.
- [13] Geoffrey Hinton, Nitish Srivastava, and Kevin Swersky. Neural networks for machine learning. Lecture 6a overview of mini-batch gradient descent course, 2012.
- [14] Bo-Yang Hsueh, Wei Li, and I-Chen Wu. Stochastic gradient descent with hyperbolic-tangent decay on classification. In *IEEE Winter Conference on Applications of Computer Vision*, pages 435–442. IEEE, 2019.
- [15] Haiwen Huang, Chang Wang, and Bin Dong. Nostalgic adam: weighting more of the past gradients when designing the adaptive learning rate. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pages 2556–2562. AAAI Press, 2019.
- [16] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations*, 2015.
- [17] Takumi Kobayashi. Phase-wise parameter aggregation for improving sgd optimization. In *IEEE Winter Conference on Applications of Computer Vision*, pages 2625–2634, 2021.
- [18] A Krizhevsky. Learning multiple layers of features from tiny images. *Master’s thesis, University of Tront*, 2009.
- [19] Ya Le and Xuan Yang. Tiny imagenet visual recognition challenge. *CS 231N*, 7:7, 2015.
- [20] Liyuan Liu, Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Jiawei Han. On the variance of the adaptive learning rate and beyond. In *Proceedings of the International Conference on Learning Representations*, 2019.
- [21] Liangchen Luo, Yuanhao Xiong, Yan Liu, and Xu Sun. Adaptive gradient methods with dynamic bound of learning rate. In *International Conference on Learning Representations*, 2018.
- [22] Sashank J Reddi, Satyen Kale, and Sanjiv Kumar. On the convergence of adam and beyond. In *Proceedings of the International Conference on Learning Representations*, 2018.
- [23] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 779–788, 2016.
- [24] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in Neural Information Processing Systems*, 28, 2015.
- [25] SK Roy, ME Paoletti, JM Haut, SR Dubey, P Kar, A Plaza, and BB Chaudhuri. Angulargrad: A new optimization technique for angular convergence of convolutional neural networks. *arXiv preprint arXiv:2105.10190*, 2021.
- [26] Sebastian Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.
- [27] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 815–823, 2015.
- [28] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015.
- [29] Korsuk Sirinukunwattana, Shan E Ahmed Raza, Yee-Wah Tsang, David RJ Snead, Ian A Cree, and Nasir M Rajpoot. Locality sensitive deep learning for detection and classification of nuclei in routine colon cancer histology images. *IEEE Transactions on Medical Imaging*, 35(5):1196–1206, 2016.
- [30] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In *Proceedings of the International Conference on Machine Learning*, pages 1139–1147, 2013.
- [31] Manzil Zaheer, Sashank Reddi, Devendra Sachan, Satyen Kale, and Sanjiv Kumar. Adaptive methods for nonconvex

optimization. In *Advances in neural information processing systems*, pages 9793–9803, 2018.

- [32] Juntang Zhuang, Tommy Tang, Sekhar Tatikonda, Nicha Dvornek, Yifan Ding, Xenophon Papademetris, and James S Duncan. Adabelief optimizer: Adapting stepsizes by the belief in observed gradients. In *Proceedings of the Conference on Neural Information Processing Systems*, 2020.
- [33] Martin Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the International Conference on Machine Learning*, pages 928–936, 2003.

## Supplementary

### A. Convergence Proof

**Lemma 1.** Let  $\eta \triangleq \frac{\beta_1^2}{\sqrt{\beta_2}}$ . For  $\beta_1, \beta_2 \in [0, 1)$  that satisfy  $\frac{\beta_1^2}{\sqrt{\beta_2}} < 1$  and bounded  $g_t, \|g_t\|_2 \leq G, \|g_t\|_\infty \leq G_\infty, e_t \leq G_\infty, \frac{e_t}{\|g_t\|_2} \leq \frac{G_\infty}{G}$ , the following inequality holds,

$$\sum_{t=1}^T \frac{\hat{m}_{t,i}^2}{\sqrt{t\hat{v}_{t,i}}} \leq \frac{2G_\infty^3}{G^2(1-\eta)^2\sqrt{1-\beta_2}} \|g_{1:T,i}\|_2$$

*Proof.* Under the assumption,  $\frac{\sqrt{1-\beta_2^T}}{(1-\beta_1^T)^2} \leq \frac{1}{(1-\beta_1)^2}$ . We can use the update rules of AdamNorm and expand the last term in the summation,

$$\begin{aligned} & \sum_{t=1}^T \frac{\hat{m}_{t,i}^2}{\sqrt{t\hat{v}_{t,i}}} \\ &= \sum_{t=1}^{T-1} \frac{\hat{m}_{t,i}^2}{\sqrt{t\hat{v}_{t,i}}} \\ & \quad + \frac{\sqrt{1-\beta_2^T}}{(1-\beta_1^T)^2} \frac{(\sum_{k=1}^T (1-\beta_1)\beta_1^{T-k} s_{k,i})^2}{\sqrt{T \sum_{j=1}^T (1-\beta_2)\beta_2^{T-j} g_{j,i}^2}} \\ &\leq \sum_{t=1}^{T-1} \frac{\hat{m}_{t,i}^2}{\sqrt{t\hat{v}_{t,i}}} \\ & \quad + \frac{\sqrt{1-\beta_2^T}}{(1-\beta_1^T)^2} \sum_{k=1}^T \frac{T((1-\beta_1)\beta_1^{T-k} s_{k,i})^2}{\sqrt{T \sum_{j=1}^T (1-\beta_2)\beta_2^{T-j} g_{j,i}^2}} \\ &\leq \sum_{t=1}^{T-1} \frac{\hat{m}_{t,i}^2}{\sqrt{t\hat{v}_{t,i}}} + \frac{\sqrt{1-\beta_2^T}}{(1-\beta_1^T)^2} \sum_{k=1}^T \frac{T((1-\beta_1)\beta_1^{T-k} s_{k,i})^2}{\sqrt{T(1-\beta_2)\beta_2^{T-k} g_{k,i}^2}} \end{aligned}$$

Further, we can simplify as,

$$\begin{aligned} & \sum_{t=1}^T \frac{\hat{m}_{t,i}^2}{\sqrt{t\hat{v}_{t,i}}} \\ &\leq \sum_{t=1}^{T-1} \frac{\hat{m}_{t,i}^2}{\sqrt{t\hat{v}_{t,i}}} + \frac{1}{(1-\beta_1)^2} \sum_{k=1}^T \frac{T((1-\beta_1)\beta_1^{T-k} s_{k,i})^2}{\sqrt{T(1-\beta_2)\beta_2^{T-k} g_{k,i}^2}} \\ &= \sum_{t=1}^{T-1} \frac{\hat{m}_{t,i}^2}{\sqrt{t\hat{v}_{t,i}}} + \frac{T}{\sqrt{T(1-\beta_2)}} \sum_{k=1}^T \frac{(\beta_1^{T-k} s_{k,i})^2}{\sqrt{\beta_2^{T-k} g_{k,i}^2}} \\ &= \sum_{t=1}^{T-1} \frac{\hat{m}_{t,i}^2}{\sqrt{t\hat{v}_{t,i}}} + \frac{T}{\sqrt{T(1-\beta_2)}} \sum_{k=1}^T \left( \frac{\beta_1^2}{\sqrt{\beta_2}} \right)^{T-k} \frac{s_{k,i}^2}{g_{k,i}} \\ &= \sum_{t=1}^{T-1} \frac{\hat{m}_{t,i}^2}{\sqrt{t\hat{v}_{t,i}}} + \frac{T}{\sqrt{T(1-\beta_2)}} \sum_{k=1}^T \eta^{T-k} \left( \frac{s_{k,i}}{\sqrt{g_{k,i}}} \right)^2 \\ &\leq \sum_{t=1}^{T-1} \frac{\hat{m}_{t,i}^2}{\sqrt{t\hat{v}_{t,i}}} \\ & \quad + \frac{T}{\sqrt{T(1-\beta_2)}} \sum_{k=1}^T \eta^{T-k} \left( \frac{\max(1, \frac{e_k}{\|g_k\|_2}) g_{k,i}}{\sqrt{g_{k,i}}} \right)^2 \end{aligned}$$

By considering the bound of  $e_k$  and  $\|g_k\|_2$ , we can rewrite the above relation as,

$$\begin{aligned} \sum_{t=1}^T \frac{\hat{m}_{t,i}^2}{\sqrt{t\hat{v}_{t,i}}} &\leq \sum_{t=1}^{T-1} \frac{\hat{m}_{t,i}^2}{\sqrt{t\hat{v}_{t,i}}} \\ & \quad + \frac{T}{\sqrt{T(1-\beta_2)}} \sum_{k=1}^T \eta^{T-k} \frac{G_\infty^2}{G^2} \|g_{k,i}\|_2 \end{aligned}$$

Similarly, after considering the upper bound of the rest of the terms in the summation, we can get as follows,

$$\begin{aligned} \sum_{t=1}^T \frac{\hat{m}_{t,i}^2}{\sqrt{t\hat{v}_{t,i}}} &\leq \frac{G_\infty^2}{G^2\sqrt{(1-\beta_2)}} \sum_{t=1}^T \frac{\|g_{t,i}\|_2}{\sqrt{t}} \sum_{j=0}^{T-t} t\eta^j \\ &\leq \frac{G_\infty^2}{G^2\sqrt{(1-\beta_2)}} \sum_{t=1}^T \frac{\|g_{t,i}\|_2}{\sqrt{t}} \sum_{j=0}^T t\eta^j \end{aligned}$$

We can obtain  $\sum_t t\eta^t < \frac{1}{(1-\eta)^2}$  for  $\eta < 1$  using the upper bound on the arithmetic-geometric series. Hence,

$$\sum_{t=1}^T \frac{\hat{m}_{t,i}^2}{\sqrt{t\hat{v}_{t,i}}} \leq \frac{G_\infty^2}{G^2(1-\eta)^2\sqrt{1-\beta_2}} \sum_{t=1}^T \frac{\|g_{k,i}\|_2}{\sqrt{t}}$$

By applying Lemma 10.3 of [16], we can get,

$$\sum_{t=1}^T \frac{\hat{m}_{t,i}^2}{\sqrt{t\hat{v}_{t,i}}} \leq \frac{2G_\infty^3}{G^2(1-\eta)^2\sqrt{1-\beta_2}} \|g_{1:T,i}\|_2$$

□

**Theorem 2.** Let the bounded gradients for function  $f_t$  (i.e.,  $\|g_{t,\theta}\|_2 \leq G$  and  $\|g_{t,\theta}\|_\infty \leq G_\infty$ ) for all  $\theta \in R^d$ . Also assume that AdamNorm produces the bounded distance between any  $\theta_t$  (i.e.,  $\|\theta_n - \theta_m\|_2 \leq D$  and  $\|\theta_n - \theta_m\|_\infty \leq D_\infty$  for any  $m, n \in \{1, \dots, T\}$ ). Let  $\eta \triangleq \frac{\beta_1^2}{\sqrt{\beta_2}}$ ,  $\beta_1, \beta_2 \in [0, 1)$  satisfy  $\frac{\beta_1^2}{\sqrt{\beta_2}} < 1$ ,  $\alpha_t = \frac{\alpha}{\sqrt{t}}$ , and  $\beta_{1,t} = \beta_1 \lambda^{t-1}$ ,  $\lambda \in (0, 1)$  with  $\lambda$  is typically close to 1, e.g.,  $1 - 10^{-8}$ . For all  $T \geq 1$ , the proposed AdamNorm optimizer shows the following guarantee:

$$\begin{aligned} R(T) &\leq \frac{D^2}{2\alpha(1-\beta_1)} \sum_{i=1}^d \sqrt{T\hat{v}_{T,i}} \\ &\quad + \frac{\alpha(1+\beta_1)G_\infty^3}{(1-\beta_1)\sqrt{1-\beta_2}(1-\eta)^2 G^2} \sum_{i=1}^d \|g_{1:T,i}\|_2 \\ &\quad + \sum_{i=1}^d \frac{D_\infty^2 G_\infty \sqrt{1-\beta_2}}{2\alpha(1-\beta_1)(1-\lambda)^2} \end{aligned}$$

*Proof.* Using Lemma 10.2 of Adam [16], we can write as

$$f_t(\theta_t) - f_t(\theta^*) \leq g_t^T(\theta_t - \theta^*) = \sum_{i=1}^d g_{t,i}(\theta_{t,i} - \theta_{*,i}^*)$$

We can write following from the AdamNorm update rule, ignoring  $\epsilon$ ,

$$\begin{aligned} \theta_{t+1} &= \theta_t - \frac{\alpha_t \hat{m}_t}{\sqrt{\hat{v}_t}} \\ &= \theta_t - \frac{\alpha_t}{(1-\beta_1^t)} \left( \frac{\beta_{1,t}}{\sqrt{\hat{v}_t}} m_{t-1} + \frac{(1-\beta_{1,t})}{\sqrt{\hat{v}_t}} g_t \right) \end{aligned}$$

where  $\beta_{1,t}$  is the  $1^{st}$  order moment coefficient at  $t^{th}$  iteration and  $\beta_1^t$  is the  $t^{th}$  power of initial  $1^{st}$  order moment coefficient.

For  $i^{th}$  dimension of parameter vector  $\theta_t \in R^d$ , we can write

$$\begin{aligned} (\theta_{t+1,i} - \theta_{*,i}^*)^2 &= (\theta_{t,i} - \theta_{*,i}^*)^2 - \frac{2\alpha_t}{1-\beta_1^t} \left( \frac{\beta_{1,t}}{\sqrt{\hat{v}_{t,i}}} m_{t-1,i} \right. \\ &\quad \left. + \frac{(1-\beta_{1,t})}{\sqrt{\hat{v}_{t,i}}} g_{t,i} \right) (\theta_{t,i} - \theta_{*,i}^*) + \alpha_t^2 \left( \frac{\hat{m}_{t,i}}{\hat{v}_{t,i}} \right)^2 \end{aligned}$$

The above equation can be reordered as

$$\begin{aligned} g_{t,i}(\theta_{t,i} - \theta_{*,i}^*) &= \frac{(1-\beta_1^t)\sqrt{\hat{v}_{t,i}}}{2\alpha_t(1-\beta_{1,t})} \left( (\theta_{t,i} - \theta_{*,i}^*)^2 \right. \\ &\quad \left. - (\theta_{t+1,i} - \theta_{*,i}^*)^2 \right) \\ &\quad + \frac{\beta_{1,t}}{1-\beta_{1,t}} (\theta_{*,i}^* - \theta_{t,i}) m_{t-1,i} \\ &\quad + \frac{\alpha_t(1-\beta_1^t)}{2(1-\beta_{1,t})} \frac{(\hat{m}_{t,i})^2}{\sqrt{\hat{v}_{t,i}}}. \end{aligned}$$

Further, it can be written as

$$\begin{aligned} &g_{t,i}(\theta_{t,i} - \theta_{*,i}^*) \\ &= \frac{(1-\beta_1^t)\sqrt{\hat{v}_{t,i}}}{2\alpha_t(1-\beta_{1,t})} \left( (\theta_{t,i} - \theta_{*,i}^*)^2 - (\theta_{t+1,i} - \theta_{*,i}^*)^2 \right) \\ &\quad + \sqrt{\frac{\beta_{1,t}}{\alpha_{t-1}(1-\beta_{1,t})}} (\theta_{*,i}^* - \theta_{t,i})^2 \sqrt{\hat{v}_{t-1,i}} \sqrt{\frac{\beta_{1,t}\alpha_{t-1}(m_{t-1,i})^2}{(1-\beta_{1,t})\sqrt{\hat{v}_{t-1,i}}}} \\ &\quad + \frac{\alpha_t(1-\beta_1^t)}{2(1-\beta_{1,t})} \frac{(\hat{m}_{t,i})^2}{\sqrt{\hat{v}_{t,i}}} \end{aligned}$$

Based on Young's inequality,  $ab \leq a^2/2 + b^2/2$  and fact that  $\beta_{1,t} \leq \beta_1$ , the above equation can be reordered as

$$\begin{aligned} g_{t,i}(\theta_{t,i} - \theta_{*,i}^*) &\leq \frac{1}{2\alpha_t(1-\beta_1)} \left( (\theta_{t,i} - \theta_{*,i}^*)^2 \right. \\ &\quad \left. - (\theta_{t+1,i} - \theta_{*,i}^*)^2 \right) \sqrt{\hat{v}_{t,i}} \\ &\quad + \frac{\beta_{1,t}}{2\alpha_{t-1}(1-\beta_{1,t})} (\theta_{*,i}^* - \theta_{t,i})^2 \sqrt{\hat{v}_{t-1,i}} \\ &\quad + \frac{\beta_1 \alpha_{t-1} (m_{t-1,i})^2}{2(1-\beta_1)\sqrt{\hat{v}_{t-1,i}}} \\ &\quad + \frac{\alpha_t}{2(1-\beta_1)} \frac{(\hat{m}_{t,i})^2}{\sqrt{\hat{v}_{t,i}}} \end{aligned}$$

We use the Lemma 1 and derive the regret bound by aggregating it across all the dimensions for  $i \in \{1, \dots, d\}$  and all the sequence of convex functions for  $t \in \{1, \dots, T\}$  in the upper bound of  $f_t(\theta_t) - f_t(\theta^*)$  as

$$\begin{aligned} R(T) &\leq \sum_{i=1}^d \frac{1}{2\alpha_1(1-\beta_1)} (\theta_{1,i} - \theta_{*,i}^*)^2 \sqrt{\hat{v}_{1,i}} \\ &\quad + \sum_{i=1}^d \sum_{t=2}^T \frac{1}{2(1-\beta_1)} (\theta_{t,i} - \theta_{*,i}^*)^2 \left( \frac{\sqrt{\hat{v}_{t,i}}}{\alpha_t} - \frac{\sqrt{\hat{v}_{t-1,i}}}{\alpha_{t-1}} \right) \\ &\quad + \frac{\beta_1 \alpha G_\infty^3}{(1-\beta_1)\sqrt{1-\beta_2}(1-\eta)^2 G^2} \sum_{i=1}^d \|g_{1:T,i}\|_2 \\ &\quad + \frac{\alpha G_\infty^3}{(1-\beta_1)\sqrt{1-\beta_2}(1-\eta)^2 G^2} \sum_{i=1}^d \|g_{1:T,i}\|_2 \\ &\quad + \sum_{i=1}^d \sum_{t=1}^T \frac{\beta_{1,t}}{2\alpha_t(1-\beta_{1,t})} (\theta_{*,i}^* - \theta_{t,i})^2 \sqrt{\hat{v}_{t,i}} \end{aligned}$$

By utilizing the assumptions that  $\alpha = \alpha_t \sqrt{t}$ ,  $\|\theta_t - \theta^*\|_2 \leq$

$D$  and  $\|\theta_m - \theta_n\|_\infty \leq D_\infty$ , we can write as

$$\begin{aligned}
R(T) &\leq \frac{D^2}{2\alpha(1-\beta_1)} \sum_{i=1}^d \sqrt{T\hat{v}_{T,i}} \\
&\quad + \frac{\alpha(1+\beta_1)G_\infty^3}{(1-\beta_1)\sqrt{1-\beta_2}(1-\eta)^2G^2} \sum_{i=1}^d \|g_{1:T,i}\|_2 \\
&\quad + \frac{D_\infty^2}{2\alpha} \sum_{i=1}^d \sum_{t=1}^T \frac{\beta_{1,t}}{(1-\beta_{1,t})} \sqrt{t\hat{v}_{t,i}} \\
&\leq \frac{D^2}{2\alpha(1-\beta_1)} \sum_{i=1}^d \sqrt{T\hat{v}_{T,i}} \\
&\quad + \frac{\alpha(1+\beta_1)G_\infty^3}{(1-\beta_1)\sqrt{1-\beta_2}(1-\eta)^2G^2} \sum_{i=1}^d \|g_{1:T,i}\|_2 \\
&\quad + \frac{D_\infty^2 G_\infty \sqrt{1-\beta_2}}{2\alpha} \sum_{i=1}^d \sum_{t=1}^T \frac{\beta_{1,t}}{(1-\beta_{1,t})} \sqrt{t}
\end{aligned}$$

It is shown in Adam [16] that  $\sum_{t=1}^T \frac{\beta_{1,t}}{(1-\beta_{1,t})} \sqrt{t} \leq \frac{1}{(1-\beta_1)(1-\eta)^2}$ . Thus, the regret bound can be written as

$$\begin{aligned}
R(T) &\leq \frac{D^2}{2\alpha(1-\beta_1)} \sum_{i=1}^d \sqrt{T\hat{v}_{T,i}} \\
&\quad + \frac{\alpha(1+\beta_1)G_\infty^3}{(1-\beta_1)\sqrt{1-\beta_2}(1-\eta)^2G^2} \sum_{i=1}^d \|g_{1:T,i}\|_2 \\
&\quad + \sum_{i=1}^d \frac{D_\infty^2 G_\infty \sqrt{1-\beta_2}}{2\alpha(1-\beta_1)(1-\lambda)^2}
\end{aligned}$$

□

## B. Algorithms

This section provides the Algorithms for different optimization techniques, including diffGrad (Algorithm 3), diffGradInject (Algorithm 4), Radam (Algorithm 5), RadamInject (Algorithm 6), AdaBelief (Algorithm 7) and AdaBeliefInject (Algorithm 8).

---

### Algorithm 3: diffGrad Optimizer

---

**Initialize:**  $\theta_0, m_0 \leftarrow 0, v_0 \leftarrow 0, t \leftarrow 0$   
**Hyperparameters:**  $\alpha, \beta_1, \beta_2$   
**While**  $\theta_t$  not converged  
 $t \leftarrow t + 1$   
 $g_t \leftarrow \nabla_\theta f_t(\theta_{t-1})$   
 $\xi_t \leftarrow 1/(1 + e^{-|g_t - g_{t-1}|})$   
 $m_t \leftarrow \beta_1 m_{t-1} + (1 - \beta_1)g_t$   
 $v_t \leftarrow \beta_2 v_{t-1} + (1 - \beta_2)g_t^2$   
**Bias Correction**  
 $\hat{m}_t \leftarrow m_t/(1 - \beta_1^t), \hat{v}_t \leftarrow v_t/(1 - \beta_2^t)$   
**Update**  
 $\theta_t \leftarrow \theta_{t-1} - \alpha \xi_t \hat{m}_t / (\sqrt{\hat{v}_t} + \epsilon)$

---



---

### Algorithm 4: diffGradNorm (diffGrad + AdaNorm) Optimizer

---

**Initialize:**  $\theta_0, m_0 \leftarrow 0, v_0 \leftarrow 0, e_0 \leftarrow 0, t \leftarrow 0$   
**Hyperparameters:**  $\alpha, \beta_1, \beta_2, \gamma$   
**While**  $\theta_t$  not converged  
 $t \leftarrow t + 1$   
 $g_t \leftarrow \nabla_\theta f_t(\theta_{t-1})$   
 $\xi_t \leftarrow 1/(1 + e^{-|g_t - g_{t-1}|})$   
 $g_{norm} \leftarrow L_2 Norm(g_t)$   
 $e_t = \gamma e_{t-1} + (1 - \gamma)g_{norm}$   
 $s_t = g_t$   
**If**  $e_t > g_{norm}$   
 $s_t = (e_t / g_{norm})g_t$   
 $m_t \leftarrow \beta_1 m_{t-1} + (1 - \beta_1)s_t$   
 $v_t \leftarrow \beta_2 v_{t-1} + (1 - \beta_2)g_t^2$   
**Bias Correction**  
 $\hat{m}_t \leftarrow m_t/(1 - \beta_1^t), \hat{v}_t \leftarrow v_t/(1 - \beta_2^t)$   
**Update**  
 $\theta_t \leftarrow \theta_{t-1} - \alpha \xi_t \hat{m}_t / (\sqrt{\hat{v}_t} + \epsilon)$

---

---

**Algorithm 5: Radam Optimizer**

---

**Initialize:**  $\theta_0, \mathbf{m}_0 \leftarrow 0, \mathbf{v}_0 \leftarrow 0, t \leftarrow 0$   
**Hyperparameters:**  $\alpha, \beta_1, \beta_2$   
**While**  $\theta_t$  not converged  
   $t \leftarrow t + 1$   
   $\mathbf{g}_t \leftarrow \nabla_{\theta} f_t(\theta_{t-1})$   
   $\mathbf{m}_t \leftarrow \beta_1 \mathbf{m}_{t-1} + (1 - \beta_1) \mathbf{g}_t$   
   $\mathbf{v}_t \leftarrow \beta_2 \mathbf{v}_{t-1} + (1 - \beta_2) \mathbf{g}_t^2$   
   $\rho_{\infty} \leftarrow 2/(1 - \beta_2) - 1$   
   $\rho_t = \rho_{\infty} - 2t\beta_2^t/(1 - \beta_2^t)$   
  **If**  $\rho_t \geq 5$   
     $\rho_u = (\rho_t - 4)(\rho_t - 2)\rho_{\infty}$   
     $\rho_d = (\rho_{\infty} - 4)(\rho_{\infty} - 2)\rho_t$   
     $\rho = \sqrt{(1 - \beta_2)\rho_u/\rho_d}$   
     $\alpha_1 = \rho\alpha/(1 - \beta_1^t)$   
    **Update**  
       $\theta_t \leftarrow \theta_{t-1} - \alpha_1 \mathbf{m}_t / (\sqrt{\mathbf{v}_t} + \epsilon)$   
  **Else**  
     $\alpha_2 = \alpha/(1 - \beta_1^t)$   
    **Update**  
       $\theta_t \leftarrow \theta_{t-1} - \alpha_2 \mathbf{m}_t$

---

---

**Algorithm 6: RadamNorm (i.e., Radam + AdaNorm) Optimizer**

---

**Initialize:**  $\theta_0, \mathbf{m}_0 \leftarrow 0, \mathbf{v}_0 \leftarrow 0, e_0 \leftarrow 0, t \leftarrow 0$   
**Hyperparameters:**  $\alpha, \beta_1, \beta_2, \gamma$   
**While**  $\theta_t$  not converged  
   $t \leftarrow t + 1$   
   $\mathbf{g}_t \leftarrow \nabla_{\theta} f_t(\theta_{t-1})$   
   $\mathbf{g}_{\text{norm}} \leftarrow L_2 \text{Norm}(\mathbf{g}_t)$   
   $e_t = \gamma e_{t-1} + (1 - \gamma) \mathbf{g}_{\text{norm}}$   
   $\mathbf{s}_t = \mathbf{g}_t$   
  **If**  $e_t > \mathbf{g}_{\text{norm}}$   
     $\mathbf{s}_t = (e_t / \mathbf{g}_{\text{norm}}) \mathbf{g}_t$   
   $\mathbf{m}_t \leftarrow \beta_1 \mathbf{m}_{t-1} + (1 - \beta_1) \mathbf{s}_t$   
   $\mathbf{v}_t \leftarrow \beta_2 \mathbf{v}_{t-1} + (1 - \beta_2) \mathbf{g}_t^2$   
   $\rho_{\infty} \leftarrow 2/(1 - \beta_2) - 1$   
   $\rho_t = \rho_{\infty} - 2t\beta_2^t/(1 - \beta_2^t)$   
  **If**  $\rho_t \geq 5$   
     $\rho_u = (\rho_t - 4)(\rho_t - 2)\rho_{\infty}$   
     $\rho_d = (\rho_{\infty} - 4)(\rho_{\infty} - 2)\rho_t$   
     $\rho = \sqrt{(1 - \beta_2)\rho_u/\rho_d}$   
     $\alpha_1 = \rho\alpha/(1 - \beta_1^t)$   
    **Update**  
       $\theta_t \leftarrow \theta_{t-1} - \alpha_1 \mathbf{m}_t / (\sqrt{\mathbf{v}_t} + \epsilon)$   
  **Else**  
     $\alpha_2 = \alpha/(1 - \beta_1^t)$   
    **Update**  
       $\theta_t \leftarrow \theta_{t-1} - \alpha_2 \mathbf{m}_t$

---

---

**Algorithm 7: AdaBelief Optimizer**

---

**Initialize:**  $\theta_0, \mathbf{m}_0 \leftarrow 0, \mathbf{v}_0 \leftarrow 0, t \leftarrow 0$   
**Hyperparameters:**  $\alpha, \beta_1, \beta_2$   
**While**  $\theta_t$  not converged  
   $t \leftarrow t + 1$   
   $\mathbf{g}_t \leftarrow \nabla_{\theta} f_t(\theta_{t-1})$   
   $\mathbf{m}_t \leftarrow \beta_1 \mathbf{m}_{t-1} + (1 - \beta_1) \mathbf{g}_t$   
   $\mathbf{v}_t \leftarrow \beta_2 \mathbf{v}_{t-1} + (1 - \beta_2) (\mathbf{g}_t - \mathbf{m}_t)^2$   
  **Bias Correction**  
     $\widehat{\mathbf{m}}_t \leftarrow \mathbf{m}_t / (1 - \beta_1^t), \widehat{\mathbf{v}}_t \leftarrow \mathbf{v}_t / (1 - \beta_2^t)$   
  **Update**  
     $\theta_t \leftarrow \theta_{t-1} - \alpha \widehat{\mathbf{m}}_t / (\sqrt{\widehat{\mathbf{v}}_t} + \epsilon)$

---

---

**Algorithm 8: AdaBeliefNorm (AdaBelief + AdaNorm) Optimizer**

---

**Initialize:**  $\theta_0, \mathbf{m}_0 \leftarrow 0, \mathbf{v}_0 \leftarrow 0, e_0 \leftarrow 0, t \leftarrow 0$   
**Hyperparameters:**  $\alpha, \beta_1, \beta_2, \gamma$   
**While**  $\theta_t$  not converged  
   $t \leftarrow t + 1$   
   $\mathbf{g}_t \leftarrow \nabla_{\theta} f_t(\theta_{t-1})$   
   $\mathbf{g}_{\text{norm}} \leftarrow L_2 \text{Norm}(\mathbf{g}_t)$   
   $e_t = \gamma e_{t-1} + (1 - \gamma) \mathbf{g}_{\text{norm}}$   
   $\mathbf{s}_t = \mathbf{g}_t$   
  **If**  $e_t > \mathbf{g}_{\text{norm}}$   
     $\mathbf{s}_t = (e_t / \mathbf{g}_{\text{norm}}) \mathbf{g}_t$   
   $\mathbf{m}_t \leftarrow \beta_1 \mathbf{m}_{t-1} + (1 - \beta_1) \mathbf{s}_t$   
   $\mathbf{v}_t \leftarrow \beta_2 \mathbf{v}_{t-1} + (1 - \beta_2) (\mathbf{g}_t - \mathbf{m}_t)^2$   
  **Bias Correction**  
     $\widehat{\mathbf{m}}_t \leftarrow \mathbf{m}_t / (1 - \beta_1^t), \widehat{\mathbf{v}}_t \leftarrow \mathbf{v}_t / (1 - \beta_2^t)$   
  **Update**  
     $\theta_t \leftarrow \theta_{t-1} - \alpha \widehat{\mathbf{m}}_t / (\sqrt{\widehat{\mathbf{v}}_t} + \epsilon)$

---