Hindawi
Computational Intelligence and Neuroscience
Volume 2022, Article ID 9337209, 13 pages
https://doi.org/10.1155/2022/9337209

Research Article

Online Learning for DNN Training: A Stochastic Block Adaptive Gradient Algorithm

Jianghui Liu,[1] **Baozhu Li**,[2] **Yangfan Zhou**,[1] **Xuhui Zhao**,[1] **Junlong Zhu**,[1] **and Mingchuan Zhang**[1]

[1]*School of Information Engineering, Henan University of Science and Technology, Luoyang 471023, China*
[2]*Internet of Things & Smart City Innovation Platform, Zhuhai Fudan Innovation Institute, Zhuhai, China*

Correspondence should be addressed to Xuhui Zhao; haustzxh@163.com

Received 31 December 2021; Revised 23 February 2022; Accepted 6 May 2022; Published 2 June 2022

Academic Editor: Friedhelm Schwenker

Copyright © 2022 Jianghui Liu et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Adaptive algorithms are widely used because of their fast convergence rate for training deep neural networks (DNNs). However, the training cost becomes prohibitively expensive due to the computation of the full gradient when training complicated DNN. To reduce the computational cost, we present a stochastic block adaptive gradient online training algorithm in this study, called SBAG. In this algorithm, stochastic block coordinate descent and the adaptive learning rate are utilized at each iteration. We also prove that the regret bound of $O(\sqrt{T})$ can be achieved via SBAG, in which T is a time horizon. In addition, we use SBAG to train ResNet-34 and DenseNet-121 on CIFAR-10, respectively. The results demonstrate that SBAG has better training speed and generalized ability than other existing training methods.

1. Introduction

Benefitting from a great many data samples and complex training model, deep learning has gained great interest in recent years and has been applied in resource allocation [1–4], signal estimation [5, 6], computer vision [7–9], and so on. However, the computing cost is very high in the training process of deep learning, which needs large amounts of training data and iteration update to obtain good model parameters. It is key to speed up model training process and improve model performance. Therefore, besides proposing new training architecture [10], designing an effective training algorithm is also important. This study focuses on the design of efficient training algorithms for deep neural networks (DNNs). In fact, many questions in practice can be modeled to be an optimization problem in general [11–13], which can be solved by employing gradient-based methods. The stochastic gradient descent (SGD) method is an effective optimization algorithm [14]. Moreover, it is easy to implement because of its simplicity and is frequently used in the training process of DNN.

Although the simplicity of stochastic gradient descents, the problem of slow convergence rate always exists. The same learning rate is not suitable for all parameter updates across the training process, especially in the case of sparse training data. For this reason, a number of training methods are presented to address this issue, for instance, AdaGrad [15], RMSProp[16], AdaDelta [17], and Adam [18]. These methods are referred as Adam-type algorithms since the adaptive learning rates are employed. Further, Adam has attained the most wide application in many deep learning training tasks, such as optimization of convolutional neural networks and recurrent neural networks [19, 20]. Despite its popularity, Adam incurs the convergence issue. For this reason, AMSGrad [21] was presented by introducing a nonincreasing learning rate. Besides, the learning rates of the Adam algorithm are either too big or too small, which results in poor generalization performance. To avoid the learning rate of extreme cases, a variant of Adam, Padam [22], was presented via employing a partial adaptive parameter p. SWATS [23] used the switch method from Adam to SGD. AdaBound [24] limited the learning rate to a dynamic bound over time at each iteration.

In deep learning, gradient-based methods are used to optimize the model parameter, which needs to calculate the gradients of all coordinates in decision vectors at each iteration, and a huge number of data and complex model lead to expensive computation cost. Randomized block coordinate descent is an efficient method for high-dimensional optimization problem and has been successfully utilized in the large-scale problem generated in machine learning [25]. It divides the set of variables into different blocks and carries out a gradient update step on a selected block coordinates randomly at each iteration, while holding the remaining ones fixed. In this way, the computational expense of each iteration can be effectively reduced.

In this study, we propose a stochastic block adaptive gradient online learning (SBAG) algorithm to rapidly train DNN, which incorporates an adaptive learning rate and stochastic block coordinate approach to improve the generalization ability and computation cost. Our key contributions are as follows:

- We present the SBAG algorithm based on the stochastic block coordinate descent method and AdaBound optimization algorithm to solve high-dimensional optimization problems.

(i) We provide the theoretical analysis on the convergence for SBAG. Moreover, we show that SBAG is convergent in the convex setting under common assumptions and its regret is bounded by $O(\sqrt{T})$, where T is the time horizon.

(ii) We demonstrate the performance of SBAG on a public dataset. The simulation results show that the algorithm takes lesser time to achieve the best accuracy on the training set and test set, and it outperforms other methods.

The rest of this study is arranged as follows. In the next two sections, we will review the extant literature and introduce related background. In Section 4, we will present SBAG in detail. In Sections 5 and 6, we will describe our convergence analysis and performance evaluation. Finally, we present the conclusion of this paper in Section 7.

2. Related Work

SGD is one of the most popular algorithms used in DNN because of its implementation easily. However, it has the same learning rate for all parameters updated at each iteration across the training process, and the parameters are updated to the same extent no matter how different the feature frequencies are, which consequently results in slow convergence rate and poor performance. Hence, some variants of SGD were proposed to improve its convergence rate by either making the learning rate adaptive or using historical gradient information for descent direction. Ghadimi et al. [26] used the Heavy-ball method to combine one-order historical gradients and current gradients for updates. Sutskever et al. [27] presented Nesterov's accelerated gradient (NAG) method. Duchi et al. [15] proposed AdaGrad that first used an adaptive learning rate, whereas AdaGrad's performance is worse in the case of dense gradients because all historical gradients are used in the updates, and this limitation is more severe when dealing with high-dimensional data in deep learning. Hinton [16] proposed RMSProp, which utilizes an exponential moving average to solve the problem that the learning rate drops sharply in AdaGrad. Zeiler [17] proposed AdaDelta, which prevents learning rate decay and gradient disappearance over time. In fact, further research was to combine adaptive learning rate with historical gradient information, such as those used in Adam [18] and AMSGrad [21]. Moreover, Adam has a good convergence rate in many scenarios. However, it was found that Adam may not converge in the later stage of the training process on account of oscillated learning rate. Reddi et al. [21] presented AMSGrad, but the result of the experiments was not much better than Adam. In general, Adam-type algorithms have better performance on convergence, but often do not work well as SGD in out of sample. To address this issue, Keskar and Socher [23] proposed the SWATS algorithm. SWATS utilizes Adam to learn in the early part of the training and switches to SGD in the later stage of the training. In this case, it enjoys the quick convergence rate of Adam and the good performance of SGD, but the switching time is difficult to determine in practice. Huang et al. [28] presented NosAdam increasing the effect of past gradients on parameter update to avoid trapping in local or divergence. Nevertheless, it depends a lot on the initial conditions. Padam [22] introduced a parameter p making the level of adaptivity of the update process controlled. Luo et al. [24] proposed the AdaBound algorithm, which provides a dynamic bound for learning rate, and AdaBound is evaluated on a public dataset and is shown to converge as fast as Adam and perform as well as SGD. However, the aforementioned methods need to calculate all coordinates of gradients in decision vectors at each iteration, and computation cost will be aggravated due to the high-dimensional data and complex model structure.

The randomized block coordinate descent method is a powerful and effective approach for the high-dimensional optimization problem. It employs randomized strategies to pick a block of variables to update per iteration. For general gradient descent algorithms, all the coordinates of gradient vector should be calculated each time. One can easily observe that this will incur significant computing cost when dealing with high-dimensional data. In contrast, the randomized block coordinate method only calculates one block coordinate of gradient vector, which is considered as the descent direction. In particular, the randomized block method selects a coordinate based on probability p and updates the responding decision variable according to its descent direction. In addition, other coordinates of decision vector remain the same as the last time. Although the randomized block coordinate method can save significant computing cost for the learner, especially in optimization problems with high dimension data, it uses the fixed learning rate that scales the entries of gradient equally, and an adaptive learning rate has not been applied in this method.

Compared with the current work, we combine the randomized block coordinate descent method with an adaptive learning rate in this study. At each iteration, a part

of gradient vectors is picked randomly, and the corresponding decision vectors are updated. In this way, the gradient is then calculated based on the chosen block coordinates instead of full gradients. Moreover, the extreme learning rates are restricted to a suitable range. Our method not only enjoys good generalization performance but also saves computation cost.

3. Preliminaries

In this section, we first introduce the optimization problem in detail. Then, we begin with the background about the randomized block coordinate method.

3.1. The Online Optimization Problem. In this work, the analysis of sequence iteration optimization problem is based on the online learning framework, which can be seen as a trade-off between a learner (the algorithm) and an opponent. In such an online convex setting, the learner selects a decision point $\mathbf{x}_t \in \mathcal{X}$ produced by the algorithm per time step $t, t = 1, \ldots, T$, and $\mathcal{X}$ is a convex and compact subset of $\mathbb{R}^n$. At the same time, the opponent responds to the decision of the learner with a loss function f_t, which is convex and unknown in advance, and the algorithm suffers a loss $f_t(\mathbf{x}_t)$. Repeating the process, we have a sequence of loss functions $\{f_1(\mathbf{x}_1), f_2(\mathbf{x}_2), \ldots, f_t(\mathbf{x}_t)\}$ where $f_t: \mathcal{X} \longrightarrow \mathbb{R}$, and they vary with time t. In general, the online learner's prediction problem can be represented as follows:

$$\min_{\mathbf{x}\in\mathcal{X}} \sum_{t=1}^{T} f_t(\mathbf{x}). \tag{1}$$

For online learning tasks, the goal is to optimize the regret R_T of the online learner's predictions against the optimal decision in hindsight, which is defined as the difference in the total sum of loss functions $\sum_{t=1}^{T} f_t(\mathbf{x}_t)$ after performing online learning over T rounds and its minimum value $\sum_{t=1}^{T} f_t(\mathbf{x}^*)$ in the deterministic decision point $\mathbf{x}^*$. In particular, we define the regret in the following:

$$R_T = \sum_{t=1}^{T} f_t(\mathbf{x}_t) - \sum_{t=1}^{T} f_t(\mathbf{x}^*), \tag{2}$$

where $\mathbf{x}^* \coloneqq \operatorname{argmin}_{\mathbf{x}\in\mathcal{X}} f_t(\mathbf{x})$, $t = 1, 2, \ldots, T$. It is desired that if the regret of online optimization algorithm is a sublinear function of T, which suggests $\lim_{T\longrightarrow\infty} R_T/T = 0$, then, on average, the online learner executes just and the fixed optimal decision afterwards. In other words, the proposed algorithm converges when its R_T is bounded. Throughout this study, the diameter of convex compact set $\mathcal{X}$ is assumed to be bounded and $\|\nabla f_t(\mathbf{x}_t)\|$ is bounded for all $t = 1, 2, \ldots, T$. Hereafter, $\|\cdot\|$ denotes the ℓ_2 norm.

3.2. Relevant Definitions. Now, we will describe the relevant definitions that are used in the next sections.

Definition 1. A function $f(\cdot): \mathcal{X} \longrightarrow \mathbb{R}$ is L-Lipschitz, where L is Lipschitz constant, and $L > 0$; if $\forall \mathbf{x}, \mathbf{y} \in \mathcal{X}$,

$$|f(\mathbf{x}) - f(\mathbf{y})| \le L\|\mathbf{x} - \mathbf{y}\|. \tag{3}$$

Definition 2 (Equation (3.2) of Section 3 in [29]) A function $f(\cdot): \mathcal{X} \longrightarrow \mathbb{R}$ is convex and differentiable where $\mathcal{X}$ is a convex set; if $\forall \mathbf{x}, \mathbf{y} \in \mathcal{X}$,

$$f(\mathbf{y}) \ge f(\mathbf{x}) + \langle \nabla f(\mathbf{x}), (\mathbf{y} - \mathbf{x}) \rangle. \tag{4}$$

Definition 3. A function $f(\cdot): \mathcal{X} \longrightarrow \mathbb{R}$ is σ-strongly convex and differentiable, $\sigma > 0$, and if $\forall \mathbf{x}, \mathbf{y} \in \mathcal{X}$,

$$f(\mathbf{y}) \ge f(\mathbf{x}) + \langle \nabla f(\mathbf{x}), (\mathbf{y} - \mathbf{x}) \rangle + \frac{\sigma}{2}\|\mathbf{x} - \mathbf{y}\|^2. \tag{5}$$

4. SBAG Algorithm and Assumptions

This section presents the proposed algorithm, followed by the common assumptions for convergence analysis of the algorithm.

4.1. Algorithm Design. In this study, we develop the high-dimensional online learning problems and aim to solve the optimization problem (1) by incorporating the stochastic block coordination method and adaptive learning rate. Because the dimensionality n of the decision variable $\mathbf{x}$ is high, the computing cost of the gradients is prohibitive. In addition, the tuning of the learning rate is challenging. For these reasons, a stochastic block coordinate adaptive optimization algorithm, dubbed SBAG, is proposed for settling the online problem (1). In our algorithm, the objective functions at different times satisfy some conditions, which are displayed in Assumption 1.

SBAG is described in Algorithm 1, whose input includes $\mathbf{x}_1 = 0$, $\mathbf{m}_1 = 0$, and $\mathbf{v}_1 = 0$. The parameters of SBAG are $\beta_{1t} \in [0, 1)$, $\beta_1 \triangleq \beta_{11}$, $\beta_2 \in [0, 1)$, and $\alpha_t = 1/\sqrt{t}$, where $t = 1, 2, \ldots, T$. At each round t, a n order diagonal matrix M_t is generated and includes random variables $\{w_{t,i}\}$ with $\mathbb{P}(w_{t,i} = 0) \coloneqq 1 - p_t$ and $\mathbb{P}(w_{t,i} = 1) \coloneqq p_t$, for $t = 0, 1, \ldots, T$ and $i = 1, \ldots, n$. In particular, the gradient $\mathbf{d}_t$ is computed as follows.

$$\mathbf{d}_t = M_t \nabla f_t(\mathbf{x}_t), \tag{6}$$

where $M_t \triangleq \operatorname{diag}\{\mathbf{w}_t\} = \operatorname{diag}\{w_{t,1}, w_{t,2}, \ldots, w_{t,n}\}$, and elements of $\mathbf{w}_t$ consist of 0 and 1. When $w_{t,i} = 1$, it means that the ith coordinate of decision vector is selected to calculate the gradient at time t. From (6), one can observe that the computation cost is greatly reduced at each iteration. In addition, let $\mathcal{H}^t$ denotes the $\sigma-$ algebra, which means $\mathcal{H}^t$ consists of all variables before time t. More explicitly, $\mathcal{H}^t = \{M_1, M_2, \ldots, M_{t-1}\}$.

By Using $\mathbf{d}_t$, one and second momentum terms $\mathbf{m}_t$ and $\mathbf{v}_t$ are obtained as follows, respectively.

$$\mathbf{m}_t = \beta_{1t}\mathbf{m}_{t-1} + (1 - \beta_{1t})\mathbf{d}_t, \tag{7}$$

$$\mathbf{v}_t = \beta_2\mathbf{v}_{t-1} + (1 - \beta_2)\mathbf{d}_t^2. \tag{8}$$

Input: $\mathbf{x}_1$
Parameter: $\mathbf{x}_1 \in \mathcal{X}$, and $\beta_{1t} \in [0,1)$ where $\beta_{11} = \beta_1$ and $\beta_2 \in [0,1)$. p_t denotes coordinate selection probability at time t. Moreover, $\beta_{1t} = \beta_1 \lambda^t$ where $\lambda \in (0,1)$ and $t = 1, 2, \ldots, T$.
Initially Set: $\mathbf{m}_1 = 0$ and $\mathbf{v}_1 = 0$.
Output: $\mathbf{x}_{t+1}$

(1) **for** $t = 1, 2, 3, \ldots$ **do**
(2) $t \leftarrow t + 1$
(3) **Generating** diagonal matrix $M_t = \mathrm{diag}\{\mathbf{w}_t\}$ with probability p_t
(4) $d_{t,i} = \begin{cases} \nabla_i f_t(x_t), & w_{t,i} = 1 \\ 0, & w_{t,i} = 0 \end{cases}$
(5) Generating gradient $\mathbf{d}_t = [d_{t,1} d_{t,2} \ldots d_{t,n}]$
(6) $\mathbf{m}_t = \beta_{1t}\mathbf{m}_{t-1} + (1 - \beta_{1t})\mathbf{d}_t$
(7) $\mathbf{v}_t = \beta_2 \mathbf{v}_{t-1} + (1 - \beta_2)\mathbf{d}_t^2$
(8) $\widehat{\mathbf{v}}_t = \max\{\mathbf{v}_t, \mathbf{v}_{t-1}\}$ and $V_t = \mathrm{diag}\{\widehat{\mathbf{v}}_t\}$
(9) $\widetilde{\mu}_t = \mathrm{Clip}\left\{\alpha/\sqrt{V_t}, \mu_{\mathrm{low}}(t), \mu_{\mathrm{upp}}(t)\right\}$
(10) $\mu_t = \widetilde{\mu}_t/\sqrt{t}$
(11) $\mathbf{x}_{t+1} = \Pi_{\mathcal{X}, \mathrm{diag}\{\mu_t^{-1}\}}(\mathbf{x}_t - \mu_t \circ \mathbf{m}_t)$
(12) **end for**
(13) **return** $\mathbf{x}_{t+1}$

Algorithm 1: SBAG.

Furthermore, SBAG introduces a bound of learning rate as follows:

$$\widetilde{\mu}_t = \mathrm{Clip}\left\{\frac{\alpha}{\sqrt{V_t}}, \mu_{\mathrm{low}}(t), \mu_{\mathrm{upp}}(t)\right\}, \tag{9}$$

where each element of the learning rate $\alpha/\sqrt{V_t}$ is clipped to constrain in an internal at time t, and the upper and lower bounds of the interval are $\mu_{\mathrm{low}}(t)$ and $\mu_{\mathrm{upp}}(t)$, respectively. That is, the output of equation (9) is constrained in $[\mu_{\mathrm{low}}(t), \mu_{\mathrm{upp}}(t)]$, and the technique was also used in [23, 24]. Moreover, let

$$\mu_t = \frac{\widetilde{\mu}_t}{\sqrt{t}}. \tag{10}$$

Then, SBAG updates $\mathbf{x}_{t+1}$ as follows:

$$\mathbf{x}_{t+1} = \Pi_{\mathcal{X}, \mathrm{diag}\{\mu_t^{-1}\}}(\mathbf{x}_t - \mu_t \circ \mathbf{m}_t), \tag{11}$$

where $\circ$ is the coordinate-wise product operator. Furthermore, the projection step of equation (11) is equivalent to the following:

$$\mathbf{x}_{t+1} = \mathrm{argmin}_{\mathbf{x} \in \mathcal{X}} \left\| \mu_t^{-1/2} \circ [\mathbf{x} - (\mathbf{x}_t - \mu_t \circ \mathbf{m}_t)] \right\|. \tag{12}$$

4.2. Assumptions. Before presenting the convergence analysis of SBAG, we will now introduce the below common assumptions.

Assumption 1. Loss functions $\{f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_t(\mathbf{x})\}$, where $t = 1, 2, \ldots, T$, are convex, differentiable, and L-Lipschitz over $\mathcal{X}$.

Assumption 2. In this study, $\mathcal{X}$ is a bounded feasible set; *i.e.*, $\|\mathbf{x}_i - \mathbf{x}_j\| \le B_\infty$, where $i, j \in \{1, 2, \ldots, T\}$ and $B_\infty > 0$.

Assumption 3. In this study, $\|\nabla f_t(\mathbf{x}_t)\|$ is bounded for all $t = 1, 2, \ldots, T$ over $\mathcal{X}$; *i.e.*, $\|\nabla f_t(\mathbf{x}_t)\| \le C_\infty$, where $C_\infty > 0$.

Assumptions 1–3 are some standard assumptions in the literature, for example [18, 21, 24]. In addition, the convergence of SBAG is analyzed based on these assumptions in the following.

5. Convergence Analysis

Now, we will analyze the convergence of SBAG. We consider the regret, equation (2), in the online optimization problem (a typical scenario). The proposed algorithm generates the gradient $\mathbf{d}_t$ with probability p_t at time t. Therefore, $\mathbf{d}_t$ is a random variable. Moreover, $\mathbf{x}_t$ is calculated by $\mathbf{d}_t$ and $\mathbf{x}_{t-1}$ at time t. According to the knowledge of probability and statistics, the expectation should be considered when the variable is randomized. Therefore, we define the regret of SBAG as follows:

$$\widehat{R}_T = \sum_{t=1}^{T} [\mathbb{E}[f_t(\mathbf{x}_t)] - f_t(\mathbf{x}^*)]. \tag{13}$$

From the convexity of f_t, it follows that

$$f_t(\mathbf{x}_t) - f_t(\mathbf{x}^*) \le \nabla f_t(\mathbf{x}_t)^\top (\mathbf{x}_t - \mathbf{x}^*). \tag{14}$$

Moreover, by the definition of matrix M_t, we know that M_t is a sparse matrix. Therefore, applying equation (14) leads to

$$f_t(\mathbf{x}_t) - f_t(\mathbf{x}^*) \le \left\| M_t \nabla f_t(\mathbf{x}_t)^\top (\mathbf{x}_t - \mathbf{x}^*) \right\| + \nabla f_t(\mathbf{x}_t)^\top (\mathbf{x}_t - \mathbf{x}^*) = \left\| \mathbf{d}_t^\top (\mathbf{x}_t - \mathbf{x}^*) \right\| + \nabla f_t(\mathbf{x}_t)^\top (\mathbf{x}_t - \mathbf{x}^*). \tag{15}$$

Taking conditional expectation (conditioned on $\mathscr{H}^t$) on both sides of equation (15), it implies that

$$\mathbb{E}\left[f_t(\mathbf{x}_t) | \mathscr{H}^t \right] - \mathbb{E}\left[f_t(\mathbf{x}^*) | \mathscr{H}^t \right] \le \mathbb{E}\left[\left\| \mathbf{d}_t^\top (\mathbf{x}_t - \mathbf{x}^*) \right\| | \mathscr{H}^t \right] + \mathbb{E}\left[\nabla f_t(\mathbf{x}_t)^\top (\mathbf{x}_t - \mathbf{x}^*) | \mathscr{H}^t \right]. \tag{16}$$

By equation (1.1f) of Section 4 in [30], and taking unconditional expectation for equation (16), it follows that

$$\mathbb{E}[f_t(\mathbf{x}_t)] - f_t(\mathbf{x}^*)] \le \mathbb{E}\left[\left\| \mathbf{d}_t^\top (\mathbf{x}_t - \mathbf{x}^*) \right\| \right] + \mathbb{E}\left[\nabla f_t(\mathbf{x}_t)^\top (\mathbf{x}_t - \mathbf{x}^*) \right]. \tag{17}$$

From equations (13) and (17), the following equation holds

$$\widehat{R}_T = \sum_{t=1}^{T} \mathbb{E}\left[\left\| \mathbf{d}_t^\top (\mathbf{x}_t - \mathbf{x}^*) \right\| \right] + \sum_{t=1}^{T} \mathbb{E}\left[\nabla f_t(\mathbf{x}_t)^\top (\mathbf{x}_t - \mathbf{x}^*) \right]. \tag{18}$$

To get the bound of $\widehat{R}_T$, we should consider the two terms on the right side of equation (18). Thus, we first propose the following lemmata to estimate term $\sum_{t=1}^{T} \mathbb{E}[\|\mathbf{d}_t^\top (\mathbf{x}_t - \mathbf{x}^*)\|]$.

Lemma 1. *If Assumptions 1 to 3 are satisfied, sequences $\{\mathbf{x}_t\}, \{\mathbf{m}_t\}$, and $\{\mathbf{v}_t\}$ are generated by SBAG with $t \in \{1, 2, \ldots, T\}$. Moreover, $\mathcal{X}$ is a convex and compact set. $\beta_1 := \beta_{11}, \beta_{1t} \le \beta_{1(t-1)} \le \beta_1$, and $\beta_1/\sqrt{\beta_2} \le 1$ for $t = 1, \ldots, T$. In addition, suppose $\mu_{low}(t+1) \ge \mu_{low}(t) \ge 0$, $\mu_{upp}(t+1) \le \mu_{upp}(t)$, and $lim_{t \longrightarrow \infty} \mu_{low}(t) = lim_{t \longrightarrow \infty} \mu_{upp}(t) = \alpha$, where $\alpha > 0$. Let $L_\infty := \mu_{low}(1)$, $U_\infty := \mu_{upp}(1)$, and $p_t \in [p_{\min}, p_{\max}]$. Then, we have the following relation:*

$$\sum_{t=1}^{T} \mathbb{E}\left[\left\| \mu_t^{1/2} \circ \mathbf{m}_t \right\|^2 \right] \le \frac{p_{\max} \beta_1 U_\infty (2\sqrt{T} - 1)}{1 - \beta_1} \sum_{i=1}^{n} \left\| d_{1:T,i} \right\|^2. \tag{19}$$

Proof. From equations (9) and (10), it follows that

$$\sqrt{t} \left\| \mu_t \right\|_\infty \le \mu_{\text{upp}}(t) \le \mu_{\text{upp}}(1) := U_\infty, \tag{20}$$

and

$$\sqrt{t} \left\| \mu_t \right\|_\infty \ge \mu_{\text{low}}(t) \ge \mu_{\text{low}}(1) := L_\infty. \tag{21}$$

From equations (20) and (21), and by property of expectation, it can be verified that

$$\begin{aligned} \sum_{t=1}^{T} \mathbb{E}\left[\left\| \mu_t^{1/2} \circ \mathbf{m}_t \right\|^2 \right] &\le \sum_{t=1}^{T} \mathbb{E}\left[\left\| \mathbf{m}_t \right\|^2 \frac{U_\infty}{\sqrt{t}} \right] = U_\infty \sum_{t=1}^{T} \frac{p_t \left\| \mathbf{m}_t \right\|^2}{\sqrt{t}} \le p_{\max} U_\infty \sum_{t=1}^{T} \frac{\left\| \mathbf{m}_t \right\|^2}{\sqrt{t}}, \\ &= p_{\max} U_\infty \sum_{t=1}^{T-1} \frac{\left\| \mathbf{m}_t \right\|^2}{\sqrt{t}} + p_{\max} U_\infty \sum_{i=1}^{n} \frac{m_{T,i}^2}{\sqrt{T}}. \end{aligned} \tag{22}$$

Plugging equation (7) into equation (22), it yields

$$\begin{aligned} &\sum_{t=1}^{T} \mathbb{E}\left[\left\| \mu_t^{1/2} \circ \mathbf{m}_t \right\|^2 \right] \\ &\le p_{\max} U_\infty \sum_{t=1}^{T-1} \frac{\left\| \mathbf{m}_t \right\|^2}{\sqrt{t}} \\ &\quad + \frac{p_{\max} U_\infty}{\sqrt{T}} \underbrace{\sum_{i=1}^{n} \left(\sum_{j=1}^{T} \left(1 - \beta_{1j}\right) \prod_{k=1}^{T-j} \beta_{1(T-k+1)} d_{j,i} \right)^2}_{(a)}. \end{aligned} \tag{23}$$

By Cauchy–Schwarz inequality, we further bound the term (a) of equation (23) and have

$$\begin{aligned} (a) &\le \sum_{i=1}^{n} \left[\sum_{j=1}^{T} \prod_{k=1}^{T-j} \beta_{1(T-k+1)} \right] \left[\sum_{j=1}^{T} \prod_{k=1}^{T-j} \beta_{1(T-k+1)} d_{j,i}^2 \right] \\ &\le \sum_{i=1}^{n} \left(\sum_{j=1}^{T} \beta_1^{T-j} \right) \left(\sum_{j=1}^{T} \beta_1^{T-j} d_{j,i}^2 \right) \\ &\le \frac{1}{1 - \beta_1} \sum_{i=1}^{n} \sum_{j=1}^{T} \beta_1^{T-j} d_{j,i}^2. \end{aligned} \tag{24}$$

The second inequality of equation (24) follows from the fact $\beta_{1k} \le \beta_1$ for all $k \in \{1, \ldots, T\}$. In addition, the third inequality of equation (24) is due to the inequality $\sum_{j=1}^{T} \beta_1^{T-j} \le 1/1 - \beta_1$. Moreover, plugging equation (24) into equation (23) leads to

$$\begin{aligned}\sum_{t=1}^{T}\mathbb{E}\left[\left\|\mu_t^{1/2}\circ\mathbf{m}_t\right\|^2\right] &\le p_{\max}U_\infty\sum_{t=1}^{T-1}\frac{\left\|\mathbf{m}_t\right\|^2}{\sqrt{t}}\\ &\quad+\frac{p_{\max}U_\infty}{(1-\beta_1)\sqrt{T}}\sum_{i=1}^{n}\sum_{j=1}^{T}\beta_1^{T-j}d_{j,i}^2\\ &\le\frac{p_{\max}U_\infty}{1-\beta_1}\sum_{t-1}^{T}\frac{1}{\sqrt{t}}\sum_{i=1}^{n}\sum_{j=1}^{t}\beta_1^{t-j}d_{j,i}^2\\ &\le\frac{p_{\max}U_\infty}{1-\beta_1}\sum_{i=1}^{n}\sum_{t-1}^{T}\frac{1}{\sqrt{t}}\sum_{j=1}^{t}\beta_1^{t-j}d_{j,i}^2\\ &\le\frac{p_{\max}U_\infty}{1-\beta_1}\sum_{i=1}^{n}\sum_{t-1}^{T}d_{t,i}^2\sum_{j=1}^{t}\frac{\beta_1^{t-j}}{\sqrt{j}}\\ &\le\frac{p_{\max}\beta_1U_\infty}{1-\beta_1}\sum_{i=1}^{n}\sum_{t-1}^{T}d_{t,i}^2\sum_{j=1}^{t}\frac{1}{\sqrt{j}}.\end{aligned}\tag{25}$$

Moreover, since $\sum_{t=1}^{T}1/\sqrt{t}\le 1+\int_1^T 1/\sqrt{t}\,dt=2\sqrt{T}-1$, and by equation (25), it follows that

$$\begin{aligned}\sum_{t=1}^{T}\mathbb{E}\left[\left\|\mu_t^{1/2}\circ\mathbf{m}_t\right\|^2\right] &\le\frac{p_{\max}\beta_1U_\infty(2\sqrt{T}-1)}{1-\beta_1}\sum_{i=1}^{n}\sum_{t-1}^{T}d_{t,i}^2\\ &\le\frac{p_{\max}\beta_1U_\infty(2\sqrt{T}-1)}{1-\beta_1}\sum_{i=1}^{n}\left\|d_{1:\,T,i}\right\|^2.\end{aligned}\tag{26}$$

Therefore, the proof of Lemma 1 is completed. Next, we introduce Lemma 2 to estimate the term $\sum_{t=1}^{T}\mathbb{E}[\|\mathbf{d}_t^\top(\mathbf{x}_t-\mathbf{x}^*)\|]$. □

Lemma 2. *If Assumptions 1 to 3 are satisfied, sequences* $\{\mathbf{x}_t\},\{\mathbf{m}_t\}$, *and* $\{\mathbf{v}_t\}$ *are generated by SBAG with* $t\in\{1,2,\ldots,T\}$. *Moreover,* $\mathcal{X}$ *is a convex and compact set.* $\beta_1:=\beta_{11},\beta_{1t}\le\beta_{1(t-1)}\le\beta_1$, *and* $\beta_1/\sqrt{\beta_2}\le 1$, *for* $t=1,\ldots,T$. *In addition, suppose* $\mu_{low}(t+1)\ge\mu_{low}(t)\ge 0$, $\mu_{upp}(t+1)\le\mu_{upp}(t)$, *and* $lim_{t\longrightarrow\infty}\mu_{low}(t)=lim_{t\longrightarrow\infty}\mu_{upp}(t)=\alpha$, *where* $\alpha>0$. *Let* $L_\infty:=\mu_{low}(1)$, $U_\infty:=\mu_{upp}(1)$, *and* $p_t\in[p_{\min},p_{\max}]$. *Then, we have the following:*

$$\begin{aligned}\sum_{t=1}^{T}\mathbb{E}\left[\left\|\mathbf{d}_t^\top(\mathbf{x}_t-\mathbf{x}^*)\right\|\right]&\le\frac{B_\infty^2L_\infty\sqrt{T}}{2(1-\beta_1)p_{\min}}+\frac{\beta_1B_\infty^2L_\infty}{2(1-\beta_1)(1-\lambda)p_{\min}}\\ &\quad+\frac{p_{\max}\beta_1U_\infty(2\sqrt{T}-1)}{(1-\beta_1)^2}\sum_{i=1}^{n}\left\|d_{1:\,T,i}\right\|^2.\end{aligned}\tag{27}$$

Proof. Let $\mathbf{x}^*:=\operatorname{argmin}_{\mathbf{x}\in\mathcal{X}}f_t(\mathbf{x})$ with $t=1,2,\ldots,T$. By equations (11) and (12), the following equation holds

$$\begin{aligned}\mathbf{x}_{t+1}&=\Pi_{\mathcal{X},\operatorname{diag}\{\mu_t^{-1}\}}(\mathbf{x}_t-\mu_t\circ\mathbf{m}_t)\\ &=\operatorname{argmin}_{\mathbf{x}\in\mathcal{X}}\left\|\mu_t^{-1/2}\circ[\mathbf{x}-(\mathbf{x}_t-\mu_t\circ\mathbf{m}_t)]\right\|.\end{aligned}\tag{28}$$

Using Lemma 3 of [31], it can be proved that

$$\begin{aligned}\left\|\mu_t^{-1/2}\circ(\mathbf{x}_{t+1}-\mathbf{x}^*)\right\|^2&\le\left\|\mu_t^{-1/2}\circ((\mathbf{x}_t-\mu_t\circ\mathbf{m}_t)-\mathbf{x}^*)\right\|^2\\ &=\left\|\mu_t^{-1/2}\circ(\mathbf{x}_t-\mathbf{x}^*)\right\|^2-2\mathbf{m}_t^\top(\mathbf{x}_t-\mathbf{x}^*)\\ &\quad+\left\|\mu_t^{1/2}\circ\mathbf{m}_t\right\|^2.\end{aligned}\tag{29}$$

Substituting equation (7) into equation (29) yields

$$\begin{aligned}\left\|\mu_t^{-1/2}\circ(\mathbf{x}_{t+1}-\mathbf{x}^*)\right\|^2&\le\left\|\mu_t^{-1/2}\circ(\mathbf{x}_t-\mathbf{x}^*)\right\|^2+\left\|\mu_t^{1/2}\circ\mathbf{m}_t\right\|^2\\ &\quad-2[\beta_{1t}\mathbf{m}_{t-1}+(1-\beta_{1t})\mathbf{d}_t]^\top(\mathbf{x}_t-\mathbf{x}^*),\\ &=\left\|\mu_t^{-1/2}\circ(\mathbf{x}_t-\mathbf{x}^*)\right\|^2+\left\|\mu_t^{1/2}\circ\mathbf{m}_t\right\|^2\\ &\quad-2\beta_{1t}\mathbf{m}_{t-1}^\top(\mathbf{x}_t-\mathbf{x}^*)-2(1-\beta_{1t})\mathbf{d}_t^\top(\mathbf{x}_t-\mathbf{x}^*).\end{aligned}\tag{30}$$

Rearranging the terms of equation (30), and by $\beta_{1t}\le\beta_{1(t-1)}$, it follows that

$$\begin{aligned}\mathbf{d}_t^\top(\mathbf{x}_t-\mathbf{x}^*)&\le\frac{\left\|\mu_t^{-1/2}\circ(\mathbf{x}_t-\mathbf{x}^*)\right\|^2}{2(1-\beta_{1t})}+\frac{\left\|\mu_t^{1/2}\circ\mathbf{m}_t\right\|^2}{2(1-\beta_{1t})}\\ &\quad-\frac{\beta_{1t}\mathbf{m}_{t-1}^\top(\mathbf{x}_t-\mathbf{x}^*)}{1-\beta_{1t}}-\frac{\left\|\mu_t^{-1/2}\circ(\mathbf{x}_{t+1}-\mathbf{x}^*)\right\|^2}{2(1-\beta_{1t})}\\ &\le\frac{\left\|\mu_t^{-1/2}\circ(\mathbf{x}_t-\mathbf{x}^*)\right\|^2-\left\|\mu_t^{-1/2}\circ(\mathbf{x}_{t+1}-\mathbf{x}^*)\right\|^2}{2(1-\beta_{1t})}+\frac{\left\|\mu_t^{1/2}\circ\mathbf{m}_t\right\|^2}{2(1-\beta_{1t})}+\frac{\beta_{1t}\mathbf{m}_{t-1}^\top(\mathbf{x}_t-\mathbf{x}^*)}{1-\beta_{1t}}.\end{aligned}\tag{31}$$

Applying Young's inequality and the Cauchy–Schwarz inequality into equation (31) leads to

$$\begin{aligned}\mathbf{d}_t^{\top}\left(\mathbf{x}_t-\mathbf{x}^*\right)\leq&\frac{\left\|\mu_t^{-1/2}\circ\left(\mathbf{x}_{\mathrm{t}}-\mathbf{x}^*\right)\right\|^2-\left\|\mu_t^{-1/2}\circ\left(\mathbf{x}_{\mathrm{t}+1}-\mathbf{x}^*\right)\right\|^2}{2\left(1-\beta_{1t}\right)}\\&+\frac{\left\|\mu_t^{1/2}\circ\mathbf{m}_{\mathrm{t}}\right\|^2}{2\left(1-\beta_{1t}\right)}+\frac{\beta_{1t}}{2\left(1-\beta_{1t}\right)}\left\|\mu_t^{1/2}\circ\mathbf{m}_{\mathrm{t}-1}\right\|^2+\frac{\beta_{1t}}{2\left(1-\beta_{1t}\right)}\left\|\mu_t^{-1/2}\circ\left(\mathbf{x}_{\mathrm{t}}-\mathbf{x}^*\right)\right\|^2.\end{aligned}\tag{32}$$

Summing equation (32) over $t\in\{1,2,\ldots,T\}$ and taking expectation on the obtained relation imply that

$$\begin{aligned}\sum_{t=1}^{T}\mathbb{E}\left[\left\|\mathbf{d}_t^{\top}\left(\mathbf{x}_t-\mathbf{x}^*\right)\right\|\right]\leq&\mathbb{E}\left[\frac{\left\|\mu_t^{-1/2}\circ\left(\mathbf{x}_{\mathrm{t}}-\mathbf{x}^*\right)\right\|^2-\left\|\mu_t^{-1/2}\circ\left(\mathbf{x}_{\mathrm{t}+1}-\mathbf{x}^*\right)\right\|^2}{2\left(1-\beta_{1t}\right)}\right]\\&+\frac{1}{2\left(1-\beta_{1t}\right)}\mathbb{E}\left[\left\|\mu_t^{1/2}\circ\mathbf{m}_{\mathrm{t}}\right\|^2+\left\|\mu_t^{1/2}\circ\mathbf{m}_{\mathrm{t}-1}\right\|^2\right]+\frac{1}{2\left(1-\beta_{1t}\right)}\mathbb{E}\left[\beta_{1t}\left\|\mu_t^{-1/2}\circ\left(\mathbf{x}_{\mathrm{t}}-\mathbf{x}^*\right)\right\|^2\right]\\\leq&\mathrm{E}\left[\frac{\left\|\mu_t^{-1/2}\circ\left(\mathbf{x}_{\mathrm{t}}-\mathbf{x}^*\right)\right\|^2-\left\|\mu_t^{-1/2}\circ\left(\mathbf{x}_{\mathrm{t}+1}-\mathbf{x}^*\right)\right\|^2}{2\left(1-\beta_1\right)}\right]+\frac{1}{2\left(1-\beta_1\right)}\mathbb{E}\left[\left\|\mu_t^{1/2}\circ\mathbf{m}_{\mathrm{t}}\right\|^2+\left\|\mu_t^{1/2}\circ\mathbf{m}_{\mathrm{t}-1}\right\|^2\right]\\&+\frac{1}{2\left(1-\beta_1\right)}\mathbb{E}\left[\beta_{1t}\left\|\mu_t^{-1/2}\circ\left(\mathbf{x}_{\mathrm{t}}-\mathbf{x}^*\right)\right\|^2\right].\end{aligned}\tag{33}$$

By Lemma 1 and equation (33), it follows from that

$$\begin{aligned}\sum_{t=1}^{T}\mathbb{E}\left[\left\|\mathbf{d}_t^{\top}\left(\mathbf{x}_t-\mathbf{x}^*\right)\right\|\right]\leq&\sum_{t=1}^{T}\mathbb{E}\left[\frac{\left\|\mu_t^{-1/2}\circ\left(\mathbf{x}_{\mathrm{t}}-\mathbf{x}^*\right)\right\|^2}{2\left(1-\beta_1\right)}\right]\leq-\sum_{t=1}^{T}\mathbb{E}\left[\frac{\left\|\mu_t^{-1/2}\circ\left(\mathbf{x}_{\mathrm{t}+1}-\mathbf{x}^*\right)\right\|^2}{2\left(1-\beta_1\right)}\right]\\&+\sum_{t=1}^{T}\frac{\mathbb{E}\left[\beta_{1t}\left\|\mu_t^{-1/2}\circ\left(\mathbf{x}_{\mathrm{t}}-\mathbf{x}^*\right)\right\|^2\right]}{2\left(1-\beta_1\right)}+\frac{p_{\max}\beta_1U_{\infty}\left(2\sqrt{T}-1\right)}{\left(1-\beta_1\right)^2}\sum_{i=1}^{n}\left\|d_{1:\,T,i}\right\|^2\\\leq&\mathbb{E}\left[\sum_{i=1}^{n}\frac{\mu_{1,i}^{-1}\left(x_{1,i}-x_i^*\right)^2}{2\left(1-\beta_1\right)}\right]+\mathbb{E}\left[\sum_{t=2}^{T}\sum_{i=1}^{n}\frac{\mu_{t,i}^{-1}\left(x_{t,i}-x_i^*\right)^2-\mu_{t-1,i}^{-1}\left(x_{t,i}-x_i^*\right)^2}{2\left(1-\beta_1\right)}\right]\\&+\sum_{t=1}^{T}\sum_{i=1}^{n}\frac{\mathbb{E}\left[\beta_{1t}\mu_{t,i}^{-1}\left(x_{t,i}-x_i^*\right)^2\right]}{2\left(1-\beta_1\right)}+\frac{p_{\max}\beta_1U_{\infty}\left(2\sqrt{T}-1\right)}{\left(1-\beta_1\right)^2}\sum_{i=1}^{n}\left\|d_{1:\,T,i}\right\|^2.\end{aligned}\tag{34}$$

Since $\mu_t=\widetilde{\mu}_t/\sqrt{t}=\alpha/\sqrt{t\widehat{\mathbf{v}}_t}$ and $\widehat{\mathbf{v}}_t=\max\{\mathbf{v}_t,\mathbf{v}_{t-1}\}$, we have $0<\mu_t\leq\mu_{t-1}$. Therefore, we further obtain $\mu_t^{-1}\geq\mu_{t-1}^{-1}$. Then, from equation (34), it can be proved that

$$\begin{aligned}\sum_{t=1}^{T}\mathbb{E}\left[\left\|\mathbf{d}_t^{\top}\left(\mathbf{x}_t-\mathbf{x}^*\right)\right\|\right]&\le\mathbb{E}\left[\sum_{i=1}^{n}\frac{\mu_{1,i}^{-1}\left(x_{1,i}-x_i^*\right)^2}{2\left(1-\beta_1\right)}\right]\\&+\mathbb{E}\left[\sum_{t=2}^{T}\sum_{i=1}^{n}\frac{\mu_{t,i}^{-1}\left(x_{t,i}-x_i^*\right)^2-\mu_{t-1,i}^{-1}\left(x_{t,i}-x_i^*\right)^2}{2\left(1-\beta_1\right)}\right]+\sum_{t=1}^{T}\sum_{i=1}^{n}\frac{\mathbb{E}\left[\beta_{1t}\mu_{t,i}^{-1}\left(x_{t,i}-x_i^*\right)^2\right]}{2\left(1-\beta_1\right)}\\&+\frac{\left(2\sqrt{T}-1\right)p_{\max^2}U_\infty C_\infty^2}{1-\beta_1}\le\mathbb{E}\left[\sum_{i=1}^{n}\frac{\mu_{T,i}^{-1}\left(x_{T,i}-x_i^*\right)^2}{2\left(1-\beta_1\right)}\right]+\sum_{t=1}^{T}\sum_{i=1}^{n}\frac{\mathbb{E}\left[\beta_{1t}\mu_{t,i}^{-1}\left(x_{t,i}-x_i^*\right)^2\right]}{2\left(1-\beta_1\right)}\\&+\frac{p_{\max}\beta_1U_\infty\left(2\sqrt{T}-1\right)}{\left(1-\beta_1\right)^2}\sum_{i=1}^{n}\left\|d_{1:\,T,i}\right\|^2.\end{aligned}\tag{35}$$

Applying Assumption 2 and property of expectation yields

$$\begin{aligned}\sum_{t=1}^{T}\mathbb{E}\left[\left\|\mathbf{d}_t^{\top}\left(\mathbf{x}_t-\mathbf{x}^*\right)\right\|\right]&\le\frac{B_\infty^2}{2\left(1-\beta_1\right)p_{\min}}\sum_{i=1}^{n}\mu_{T,i}^{-1}\\&+\frac{B_\infty^2}{2\left(1-\beta_1\right)p_{\min}}\sum_{t=1}^{T}\sum_{i=1}^{n}\beta_{1t}\mu_{t,i}^{-1}+\frac{p_{\max}\beta_1U_\infty\left(2\sqrt{T}-1\right)}{\left(1-\beta_1\right)^2}\sum_{i=1}^{n}\left\|d_{1:\,T,i}\right\|^2\\&\le\frac{B_\infty^2L_\infty\sqrt{T}}{2\left(1-\beta_1\right)p_{\min}}+\frac{\beta_1B_\infty^2L_\infty}{2\left(1-\beta_1\right)\left(1-\lambda\right)p_{\min}}+\frac{p_{\max}\beta_1U_\infty\left(2\sqrt{T}-1\right)}{\left(1-\beta_1\right)^2}\sum_{i=1}^{n}\left\|d_{1:\,T,i}\right\|^2.\end{aligned}\tag{36}$$

Therefore, the proof of Lemma 2 is completed. Next, we estimate the last term in (18). □

Lemma 3. *If Assumptions 1 to 3 are satisfied, sequences* $\{\mathbf{x}_t\},\{\mathbf{m}_t\}$, *and* $\{\mathbf{v}_t\}$ *are generated by SBAG with* $t\in\{1,2,\ldots,T\}$. *Moreover,* $\mathcal{X}$ *is a convex and compact set. Let* $\beta_1:=\beta_{11},\beta_{1t}\le\beta_1$ *and* $\beta_1/\sqrt{\beta_2}\le1$, *for* $t=1,\ldots,T$. *In addition, suppose* $\mu_{low}(t+1)\ge\mu_{low}(t)\ge0,\mu_{upp}(t+1)\le\mu_{upp}(t)$, *and* $lim_{t\longrightarrow\infty}\mu_{low}(t)=lim_{t\longrightarrow\infty}\mu_{upp}(t)=\alpha$, *where* $\alpha>0$. *Let* $L_\infty:=\mu_{low}(1)$ *and* $U_\infty:=\mu_{upp}(1)$. *Then, we attain the following inequality:*

$$\begin{aligned}\sum_{t=1}^{T}\mathbb{E}\left[\nabla f_t\left(\mathbf{x}_t\right)^{\top}\left(\mathbf{x}_t-\mathbf{x}^*\right)\right]&\le\frac{B_\infty^2L_\infty\sqrt{T}}{2\left(1-\beta_1\right)}\\&+\frac{\beta_1B_\infty^2L_\infty}{2\left(1-\beta_1\right)\left(1-\lambda\right)}+\frac{\beta_1U_\infty\left(2\sqrt{T}-1\right)}{\left(1-\beta_1\right)^2}\sum_{i=1}^{n}\left\|d_{1:\,T,i}\right\|^2.\end{aligned}\tag{37}$$

Proof. For the original full gradient, we have $\mathbb{E}[\nabla f_t(\mathbf{x}_t)]=\nabla f_t(\mathbf{x}_t)$. Let $\mathbf{m}_t'=\beta_{1t}\mathbf{m}_{t-1}'+(1-\beta_{1t})\nabla f_t(\mathbf{x}_t)$, $\mathbf{v}_t'=\beta_2\mathbf{v}_{t-1}'+(1-\beta_2)\nabla f_t^2(\mathbf{x}_t)$, and $\mu_t'=\widetilde{\mu}_t'/\sqrt{t}$, which are generated by AdaBound [24].

The proof of Lemma 3 is similar to that of Theorem 4 in [24]. Starting with the following inequality implies

$$\begin{aligned}\sum_{t=1}^{T}\mathbb{E}\left[\left\|\nabla f_t(\mathbf{x}_t)^{\top}(\mathbf{x}_t-\mathbf{x}^*)\right\|\right]&\leq\frac{\left\|{\mu'}_t^{-1/2}\circ(\mathbf{x}_t-\mathbf{x}^*)\right\|^2-\left\|{\mu'}_t^{-1/2}\circ(\mathbf{x}_{t+1}-\mathbf{x}^*)\right\|^2}{2(1-\beta_1)}\\&+\frac{\left\|{\mu'}_t^{1/2}\circ\mathbf{m}_t'\right\|^2+\left\|{\mu'}_t^{1/2}\circ\mathbf{m}_{t-1}'\right\|^2}{2(1-\beta_1)}+\frac{\beta_{1t}\left\|{\mu'}_t^{-1/2}\circ(\mathbf{x}_t-\mathbf{x}^*)\right\|^2}{2(1-\beta_1)}\\&\leq\frac{\left\|{\mu'}_t^{-1/2}\circ(\mathbf{x}_t-\mathbf{x}^*)\right\|^2-\left\|{\mu'}_t^{-1/2}\circ(\mathbf{x}_{t+1}-\mathbf{x}^*)\right\|^2}{2(1-\beta_1)}+\frac{\beta_{1t}\left\|{\mu'}_t^{-1/2}\circ(\mathbf{x}_t-\mathbf{x}^*)\right\|^2}{2(1-\beta_1)}\\&+\frac{\beta_1U_\infty(2\sqrt{T}-1)}{(1-\beta_1)^2}\sum_{i=1}^{n}\left\|d_{1:T,i}\right\|^2\leq\frac{B_\infty^2L_\infty\sqrt{T}}{2(1-\beta_1)}+\frac{\beta_1B_\infty^2L_\infty}{2(1-\beta_1)(1-\lambda)}+\frac{\beta_1U_\infty(2\sqrt{T}-1)}{(1-\beta_1)^2}\sum_{i=1}^{n}\left\|d_{1:T,i}\right\|^2.\end{aligned}\tag{38}$$

Therefore, the proof of Lemma 3 is finished.

To attain the bound of regret $\widehat{R}_T$ in equation (18), we establish Theorem 1 as follows. □

Theorem 1. *Suppose that Assumptions 1 to 3 are satisfied, and sequences* $\{\mathbf{x}_t\},\{\mathbf{m}_t\}$, *and* $\{\mathbf{v}_t\}$ *are generated by SBAG with* $t\in\{1,2,\ldots,T\}$. *Moreover,* $\mathcal{X}$ *is a convex and compact set.* $\beta_1:=\beta_{11},\beta_{1t}\leq\beta_1$, *and* $\beta_1/\sqrt{\beta_2}\leq1$ *for* $t=1,\ldots,T$. *In addition, suppose* $\mu_{low}(t+1)\geq\mu_{low}(t)\geq0$, $\mu_{upp}(t+1)\leq\mu_{upp}(t)$, *and* $lim_{t\longrightarrow\infty}\mu_{low}(t)=lim_{t\longrightarrow\infty}\mu_{upp}(t)=\alpha$, *where* $\alpha>0$. *Let* $L_\infty:=\mu_{low}(1)$, $U_\infty:=\mu_{upp}(1)$, *and* $p_t\in[p_{\min},p_{\max}]$. *We obtain the bound of regret as follows:*

$$\begin{aligned}\widehat{R}_T\leq&\frac{(1+p_{\min})B_\infty^2L_\infty\sqrt{T}}{2(1-\beta_1)p_{\min}}+\frac{\beta_1(1+p_{\min})B_\infty^2L_\infty}{2(1-\beta_1)(1-\lambda)p_{\min}}\\&+\frac{(1+p_{\max})\beta_1U_\infty(2\sqrt{T}-1)}{(1-\beta_1)^2}\sum_{i=1}^{n}\left\|d_{1:T,i}\right\|^2.\end{aligned}\tag{39}$$

Proof. Applying lemmata 1, 2, and 3 into (18) yields

$$\begin{aligned}\widehat{R}_T\leq&\frac{B_\infty^2L_\infty\sqrt{T}}{2(1-\beta_1)p_{\min}}+\frac{\beta_1B_\infty^2L_\infty}{2(1-\beta_1)(1-\lambda)p_{\min}}\\&+\frac{p_{\max}\beta_1U_\infty(2\sqrt{T}-1)}{(1-\beta_1)^2}\sum_{i=1}^{n}\left\|d_{1:T,i}\right\|^2+\frac{B_\infty^2L_\infty\sqrt{T}}{2(1-\beta_1)}\\&+\frac{\beta_1B_\infty^2L_\infty}{2(1-\beta_1)(1-\lambda)}+\frac{\beta_1U_\infty(2\sqrt{T}-1)}{(1-\beta_1)^2}\sum_{i=1}^{n}\left\|d_{1:T,i}\right\|^2\\=&\frac{(1+p_{\min})B_\infty^2L_\infty\sqrt{T}}{2(1-\beta_1)p_{\min}}+\frac{\beta_1(1+p_{\min})B_\infty^2L_\infty}{2(1-\beta_1)(1-\lambda)p_{\min}}\\&+\frac{(1+p_{\max})\beta_1U_\infty(2\sqrt{T}-1)}{(1-\beta_1)^2}\sum_{i=1}^{n}\left\|d_{1:T,i}\right\|^2.\end{aligned}\tag{40}$$

Therefore, we complete the proof of Theorem 1.

From Theorem 1, we obtain $\lim_{T\longrightarrow\infty}\widehat{R}_T/T=0$. This suggests that SBAG is convergent. In addition, the bound of regret $\widehat{R}_T$ is $O(\sqrt{T})$; i.e., given some accuracy ε, it requires an order of $\mathcal{O}(1/\varepsilon^2)$ iterations at least to achieve the given accuracy. □

6. Performance Evaluation

In this section, we perform our experiments on a public dataset to evaluate the performance of algorithm objectively. We consider the machine learning problem, multi-classification tasks taking advantage of the DNN for the experiments.

6.1. Setup. To assess our SBAG algorithm, we research the performance on the classification task problem. We use the CIFAR-10 [32] dataset for our experiments, which is widely used for classification problem. It consists of 10 classes and 50000 training samples and 10000 test samples.

For the experiments, we use the convolutional neural network to solve classification tasks on the CIFAR-10 image dataset, which has a good effect on image classification and object recognition, and specifically implement ResNet-34 [33] and DenseNet-121 [34].

6.2. Parameters. To study the performance of our proposed algorithm, we compare SBAG with SGD [14], AdaGrad [15], and AdaBound [24]. The hyper-parameters of these algorithms are initialized as follows.

For SGD, the scale of the learning rate is selected from the set $\{100,10,1,0.1,0.01\}$. AdaGrad uses the initialized learning rate set $\{5e-2,1e-2,5e-3,1e-3,5e-4\}$, and the value 0 is set for the initial accumulator value of AdaGrad. The value of hyper-parameters of AdaBound is set the same as Adam. We directly use the initialized hyper-parameter values of AdaBound in our algorithm. In addition, we set the probability of choosing a coordinate from these values in the set $\{0.10\%,0.50\%,1.00\%,5.00\%,10.00\%,50.00\%\}$.

In addition, we define the dynamic bound functions following with [24] for our simulation experiments, i.e.,

$$\mu_{\text{low}}(t)=0.1-\frac{0.1}{(1-\beta_2)t+1},\tag{41}$$

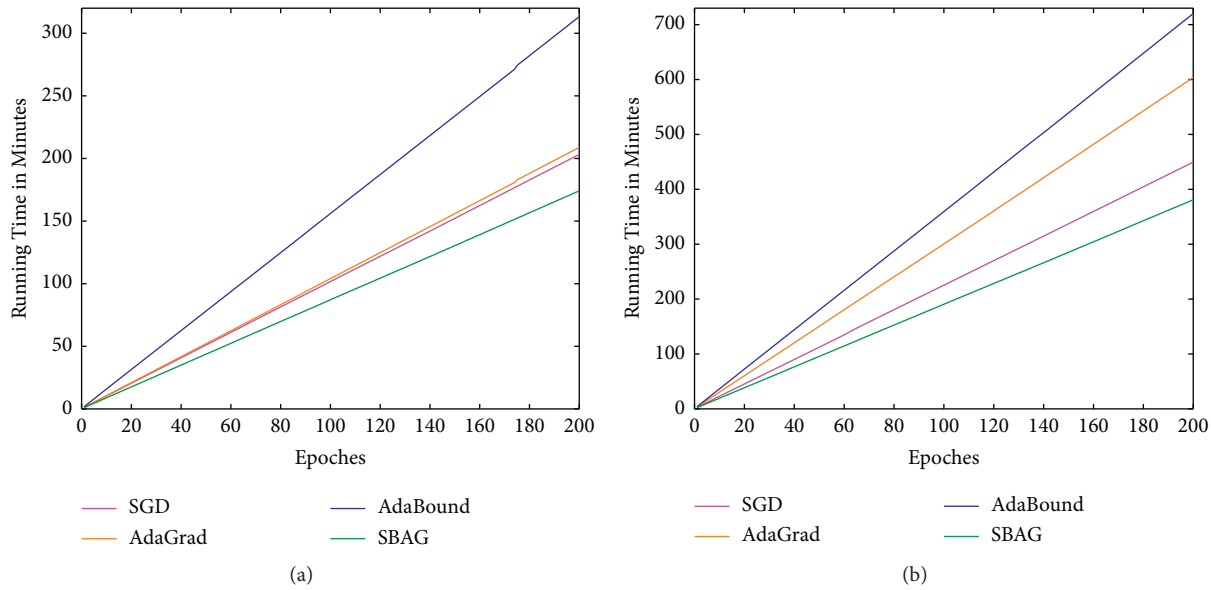

Figure 1: Running time with epochs for ResNet-34 and DenseNet-121 on CIFAR-10: a comparative summary. (a) Runtime for ResNet-34. (b) Runtime for DenseNet-121.

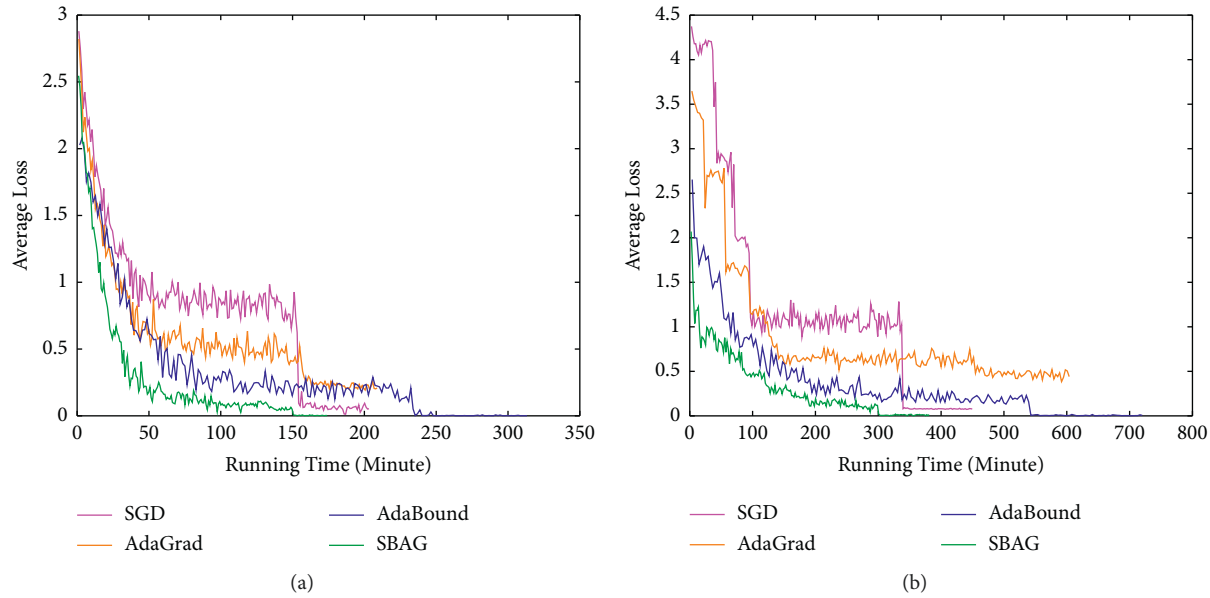

Figure 2: Loss with running time for ResNet-34 and DenseNet-121 on CIFAR-10: a comparative summary. (a) Loss for ResNet-34. (b) Loss for DenseNet-121.

and

$$\mu_{\text{upp}}(t) = 0.1 + \frac{0.1}{(1-\beta_2)t}. \tag{42}$$

6.3. Results. We take account of the image multi-class classification problem on the CIFAR-10 dataset using ResNet-34 and DenseNet-121 and run 200 epoch in this experiment. First, we operate a group of experiments with epochs and runtime for ResNet-34 and DenseNet-121 on CIFAR-10. The findings of experiments are reported in Figure 1, and when completing the same number of iterations of 200 epochs, our method takes the least time, and the AdaBound spends the most time. The main reason is that only several blocks of coordinates are calculated in the gradient descent process for our algorithm at each iteration t, while the compared algorithms calculate the full gradients at each iteration. Moreover, AdaBound combines the first- and second-order momentum, while SGD and AdaGrad only use first-order gradients; thus, SGD and AdaGrad incur less time than AdaBound. The

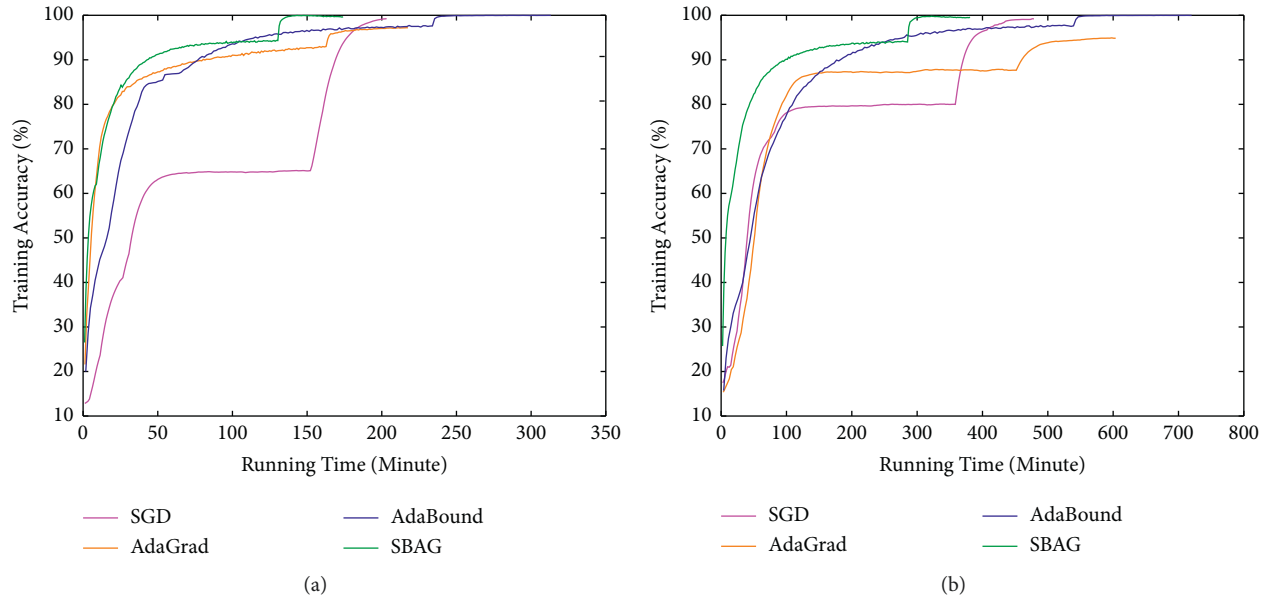

Figure 3: Training accuracy with running time for ResNet-34 and DenseNet-121 on CIFAR-10: a comparative summary. (a) Training accuracy for ResNet-34. (b) Training accuracy for DenseNet-121.

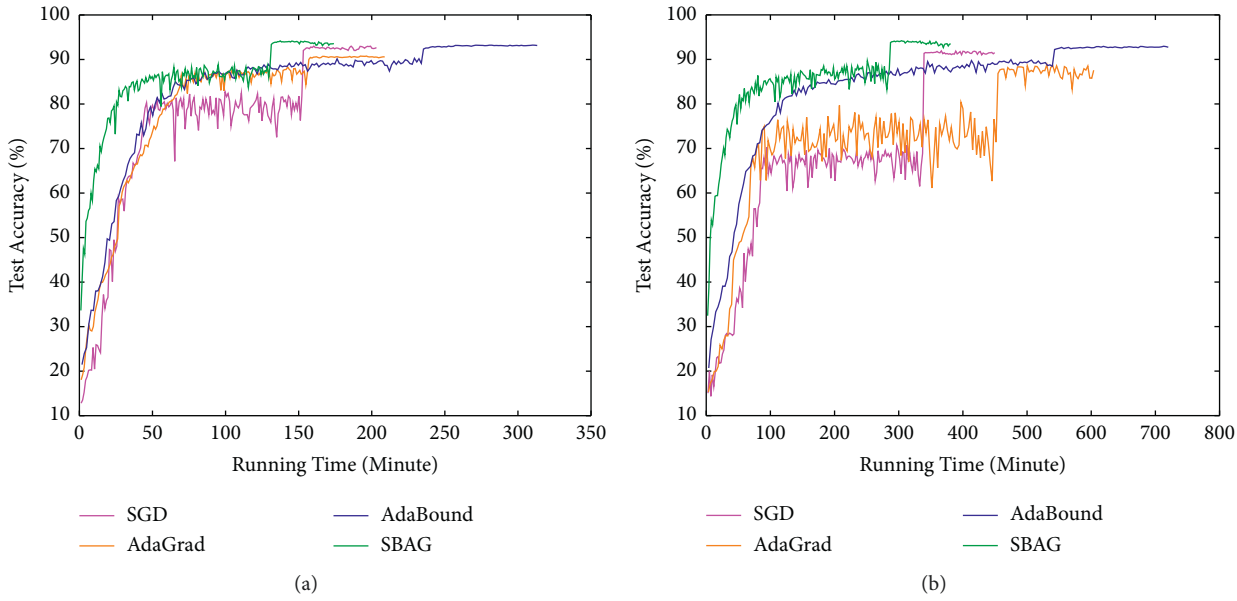

Figure 4: Test accuracy with running time for ResNet-34 and DenseNet-121 on CIFAR-10: a comparative summary. (a) Test accuracy for ResNet-34. (b) Test accuracy for DenseNet-121.

same results can be seen for the DenseNet-121 in Figure 1(b).

We present another group of experiments with average loss and running time, which are executed for ResNet-34 and DenseNet-121 on CIFAR-10. The findings are shown in Figure 2. At about 150 epochs, SGD has the biggest average loss than others and decreases sharply after that time, while the average loss of SBAG is smaller compared with others and reaches the minimum value in the shortest running time finally. The reason for fast descent rate of SBAG is due to the randomized block method, which chooses one block coordinate of decision vector to calculate the gradient. In other words, SBAG calculates more samples than other compared algorithms in the same running time. Therefore, the convergence of SBAG is verified by the findings presented in Figure 2.

In Figures 3 and 4, the training and test accuracy with running time of four algorithms are evaluated. As we can see,

in about 150 epochs, AdaBound achieves the highest accuracy, and AdaGrad and our algorithm almost have the same accuracy of 92.36% and 93.99%. As the running time goes, the AdaBound and SBAG have the accuracy of 99.96% and 99.93%, respectively. The similar results can be seen on the DenseNet-121. In a word, SBAG works well on training or test set, and at the same time, it has the good generalization ability on both ResNet-34 and DenseNet-121.

From the experiments above, we observe that the SBAG shows a very good performance on both ResNet-34 and DenseNet-121. It incurs less computation cost for each iteration in experiments, which is consistent with theory.

7. Conclusion

In this study, we proposed a randomized block adaptive gradient online learning algorithm. The proposed algorithm, SBAG, is designed to reduce the gradient computation cost of high-dimensional decision vector. The convergence analysis of SBAG and evaluations on CIFAR-10 demonstrated that the regret bound of SBAG is $O(\sqrt{T})$ when loss functions are convex and achieved significant computation cost savings, without adversely affecting the performance of the optimizer. In the same 200 epochs, the proposed algorithm has the least running time and tightly less in average loss in the end. The accuracy of training sample for ResNet-34 and DenseNet-121 is 99.93% and 99.72%, slightly less compared with that of 99.96% of AdaBound, but our method reaches the highest accuracy on the test sample than AdaBound, SGD, and AdaGrad; i.e., SBAG is the fastest in four methods, and the curves are milder than SGD.

Data Availability

The data that support the findings of this study are CIFAR-10, which is available from [32].

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this study.

Authors' Contributions

Jianghui Liu and Baozhu Li contributed equally.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China (NSFC), under Grant nos. 61976243 and 61871430, the Leading Talents of Science and Technology in the Central Plain of China, under Grant no. 214200510012, the Scientific and Technological Innovation Team of Colleges and Universities in Henan Province, under Grant no. 20IRTSTHN018, the Basic Research Projects in the University of Henan Province, under Grant no. 19zx010, the Key Scientific Research Projects of Colleges and Universities in Henan Province, under Grant no. 22A520005, the National Natural Science Foundation of China, under Grant no. 61901191, and the Shandong Provincial Natural Science Foundation, under Grant no. ZR2020LZH005.

References

[1] D. Huang, Y. Gao, Y. Li et al., "Deep learning based cooperative resource allocation in 5g wireless networks," *Mobile Networks and Applications*, Springer, Berlin, Germany, 2018.

[2] R. Dong, C. She, W. Hardjawana, Y. Li, and B. Vucetic, "Deep learning for radio resource allocation with diverse quality-of-service requirements in 5g," 2020, https://arxiv.org/abs/2004.00507.

[3] J. Qian, K. Zhu, R. Wang, and Y. Zhao, "Optimal auction for resource allocation in wireless virtualization: a deep learning approach," in *Proceedings of the 25th IEEE International Conference on Parallel and Distributed Systems, ICPADS 2019*, pp. 535–538, IEEE, Tianjin, China, December 2019.

[4] J. Chen, K. Li, K. Li, P. S. Yu, and Z. Zeng, "Dynamic planning of bicycle stations in dockless public bicycle-sharing system using gated graph neural network," *ACM Transactions on Intelligent Systems and Technology*, vol. 12, no. 2, pp. 1–22, 2021.

[5] A. Saha, V. Minz, S. Bonela, S. R. Sreeja, R. Chowdhury, and D. Samanta, "Classification of EEG signals for cognitive load estimation using deep learning architectures," in *Proceedings of the Intelligent Human Computer Interaction - 10th International Conference, IHCI 2018*, U. S. Tiwary, Ed., pp. 59–68, Springer, Allahabad, India, December 2018.

[6] X. Chang, G. Li, L. Tu, G. Xing, and T. Hao, "Deepheart: accurate heart rate estimation from PPG signals based on deep learning," in *Proceedings of the 16th IEEE International Conference on Mobile Ad Hoc and Sensor Systems, MASS*, pp. 371–379, IEEE, Monterey, CA, USA, November 2019.

[7] A. Ioannidou, E. Chatzilari, S. Nikolopoulos, and I. Kompatsiaris, "Deep learning advances in computer vision with 3d data: a survey," *ACM Computing Surveys*, vol. 50, pp. 21–38, 2017.

[8] J. Guo, H. He, T. He et al., "Gluoncv and gluonnlp: deep learning in computer vision and natural language processing," *Journal of Machine Learning Research*, vol. 21, no. 23, pp. 1–7, 2020.

[9] B. Pu, K. Li, S. Li, and N. Zhu, "Automatic fetal ultrasound standard plane recognition based on deep learning and IIoT," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 11, pp. 7771–7780, 2021.

[10] J. Chen, K. Li, K. Bilal, X. Zhou, K. Li, and P. S. Yu, "A Bi-layered parallel training architecture for large-scale convolutional neural networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 30, no. 5, pp. 965–976, 2019.

[11] F. Song, Y. Zhou, L. Chang, and H. Zhang, "Modeling space-terrestrial integrated networks with smart collaborative theory," *IEEE Netw*, vol. 33, pp. 51–57, 2018.

[12] F. Song, M. Zhu, Y. Zhou, I. You, and H. Zhang, "Smart collaborative tracking for ubiquitous power iot in edge-cloud interplay domain," *IEEE Internet of Things Journal*, vol. 7, no. 7, pp. 6046–6055, 2020.

[13] F. Song, Z. Ai, Y. Zhou, I. You, K.-K. R. Choo, and H. Zhang, "Smart collaborative automation for receive buffer control in multipath industrial networks," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 2, pp. 1385–1394, 2020.

[14] H. Robbins and S. Monro, "A stochastic approximation method," *The Annals of Mathematical Statistics*, vol. 22, no. 3, pp. 400–407, 1951.

[15] J. C. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *Journal of Machine Learning Research*, vol. 12, pp. 2121–2159, 2011.

[16] G. Hinton, "Divide the gradient by a running average of its recent magnitude," *Neural Networks for Machine Learning*, Coursera, California, USA, 2011.

[17] M. D. Zeiler, "ADADELTA: an adaptive learning rate method," 2012, https://arxiv.org/abs/1212.5701.

[18] D. P. Kingma and J. Ba, "Adam: a method for stochastic optimization," in *Proceedings of the 3rd International Conference on Learning Representations, ICLR 2015*, San Diego, CA, USA, May 2015.

[19] J. Zhang, L. Cui, and F. B. Gouza, "GADAM: genetic-evolutionary ADAM for deep neural network optimization," 2018, https://arxiv.org/abs/1805.07500.

[20] J. Bai and J. Zhang, "Bgadam: boosting based genetic-evolutionary Adam for convolutional neural network optimization," 2019, https://arxiv.org/abs/1908.08015.

[21] S. J. Reddi, S. Kale, and S. Kumar, "On the convergence of Adam and beyond," in *Proceedings of the 6th International Conference on Learning Representations ICLR 2018*, OpenReview.net, Vancouver, BC, Canada, April 2018.

[22] J. Chen, D. Zhou, Y. Tang, Z. Yang, Y. Cao, and Q. Gu, "Closing the generalization gap of adaptive gradient methods in training deep neural networks," in *Proceedings of the 29th International Joint Conference on Artificial Intelligence, IJCAI 2020*, C. Bessiere, Ed., pp. 3267–3275, ijcai.org, 2020.

[23] N. S. Keskar and R. Socher, "Improving generalization performance by switching from Adam to SGD," 2017, https://arxiv.org/abs/1712.07628.

[24] L. Luo, Y. Xiong, Y. Liu, and X. Sun, "Adaptive gradient methods with dynamic bound of learning rate," in *Proceedings of the 7th International Conference on Learning Representations, ICLR 2019*, OpenReview.net, New Orleans, LA, USA, May 2019.

[25] S. Shalev-Shwartz and A. Tewari, "Stochastic methods for l1-regularized loss minimization," *Journal of Machine Learning Research*, vol. 12, pp. 1865–1892, 2011.

[26] E. Ghadimi, H. R. Feyzmahdavian, and M. Johansson, "Global convergence of the heavy-ball method for convex optimization," in *Proceedings of the European Control Conference, ECC 2015*, pp. 310–315, IEEE, Linz, Austria, July 2015.

[27] I. Sutskever, J. Martens, G. E. Dahl, and G. E. Hinton, "On the importance of initialization and momentum in deep learning," in *Proceedings of the 30th International Conference on Machine Learning*, pp. 1139–1147, ICML, Atlanta, GA, USA, June 2013.

[28] H. Huang, C. Wang, and B. 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, IJCAI 2019*, S. Kraus, Ed., pp. 2556–2562, ijcai.org, Macao, China, August 2019.

[29] S. Boyd and L. Vandenberghe, *Convex Optimization*, Cambridge University Press, Cambridge, UK, 2004.

[30] P. Embrechts, *Probability: Theory and Examples*, University in Cambridge, Cambridge, UK, 3rd edition, 2007.

[31] H. B. McMahan and M. J. Streeter, "Adaptive bound optimization for online convex optimization," in *Proceedings of the COLT 2010 - The 23rd Conference on Learning Theory*, A. T. Kalai and M. Mohri, Eds., Omnipress, Haifa, Israel, pp. 244–256, June 2010.

[32] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," *Technical Report TR-2009*, University of Toronto, Toronto, Canada, 2009.

[33] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016*, pp. 770–778, IEEE Computer Society, Las Vegas, NV, USA, June 2016.

[34] G. Huang, Z. Liu, and K. Q. Weinberger, "Densely connected convolutional networks," 2016, https://arxiv.org/abs/1608.06993.