Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 8533687, 12 pages
https://doi.org/10.1155/2022/8533687

Research Article

A Projection-Free Adaptive Momentum Optimization Algorithm for Mobile Multimedia Computing

Lin Wang,[1] **Yangfan Zhou**,[1,2] **Xin Wang**,[3] **Zhihang Ji**,[1] **and Xin Liu**[2]

[1]*School of Information Engineering, Henan University of Science and Technology, Luoyang 471023, China*
[2]*Suzhou Institute of Nano-Tech and Nano-Bionics (SINANO), Chinese Academy of Sciences, Suzhou 215123, China*
[3]*School of Business and Management, Shanghai International Studies University, Shanghai 200083, China*

Correspondence should be addressed to Xin Wang; wangxin@shisu.edu.cn

Received 28 January 2022; Accepted 29 March 2022; Published 20 April 2022

Academic Editor: Xun Shao

In mobile multimedia applications, deep learning has received significant interest. Due to the limited computation and storage resources of mobile devices, however, general training methods are hardly suited for mobile multimedia computing. For this reason, we propose an adaptive momentum training (FWAdaBound) algorithm to reduce computation and storage cost, where the Frank-Wolfe method is employed. Furthermore, we rigorously prove the regret bound in order that $O(T^{3/4})$ can be achieved, where T is a time horizon. Finally, the convergence, cost-reduction, and generalization ability of FWAdaBound are validated through various experiments on public datasets.

1. Introduction

In mobile multimedia applications [1, 2], deep learning is a significant method for multimedia computing [3]. Meanwhile, various deep learning models have been successfully implemented in many important fields, such as convolutional neural network [4, 5], recurrent neural network [6, 7], deep belief networks [8, 9], and industrial internet applications [10]. In order to implement deep learning models in mobile multimedia, the training of deep neural networks is a crucial technology. Moreover, because the computation and storage resources of each mobile device is limited, general training methods of deep neural networks are hardly adapted to mobile multimedia computing. For this reason, how to train rapidly deep learning models with lower computational cost is one of challenging tasks in mobile multimedia applications.

In fact, the training process of deep neural networks can be regarded as an optimization process. For this reason, the design of optimization algorithms is necessary in the training process. Currently, stochastic gradient descent (SGD) is a dominative algorithm for training deep networks. SGD is applied widely over the past years since its good generalization ability and could be implemented easily. Despite SGD having performed well in some applications, however, it converges slowly. To accelerate the convergence of SGD, many researchers have proposed various adaptive momentum algorithms based on gradient descent. Generally, optimizing step size and gradient direction of SGD are two main directions that have been studied by researchers.

SGD often oscillates around the optimal solution when step sizes are fixed. To address this issue, some novel algorithms with adaptive step size have been proposed. AdaGrad [11], RMSProp [12], and Adadelta [13] make step sizes changed adaptively as training process goes on. Besides, the current gradient direction in each iteration is randomly selected, thereby it cannot find the direction to reach the optimal solution in the shortest time. For this reason, historical gradient information has been used to adjust the current gradient direction in many novel algorithms. Moreover, these algorithms often use the first-order momentum to maintain historical gradient information and the second-order momentum to adaptive step size at the same time (Adam [14], AMSGrad [15], and AdaBound [16]). It is gen-

erally believed that algorithms combining first-order and second-order momentum originated from Adam and are Adam-type algorithms.

Being an Adam-type algorithms, AdaBound not only inherits good generalization ability of SGD but also maintains fast convergence rate of Adam. However, like other Adam-type algorithm else, AdaBound also uses higher-order projection operators to handle the case where the iteration point is not within the feasible region. Since the projection operation includes the second-order Euclidean distance calculation or higher-order methods measurement, it has a large computational cost in each iteration. Therefore, algorithms with projection operations like AdaBound become prohibitive when dealing with large-scale problems including massive high-dimensional data. To tackle this problem, we focus on proposing a projection-free algorithm based on AdaBound. In the field of optimization, the Frank-Wolfe method is one of the projection-free technologies, which is commonly used in replacing high-order projection operators with linear searches. Therefore, in this paper, we redesign AdaBound algorithm, which is called FWAdaBound by using the Frank-Wolfe method to reduce computation cost of AdaBound. Moreover, we prove that the FWAdaBound algorithm converges under convex conditions and attain a guaranteed regret bound related to the sublinear correlation of time horizon. In addition, FWAdaBound successfully retains AdaBound's performance on convergence and generalization ability.

In this paper, the summary of our contributions is presented as follows:

(i) We propose the FWAdaBound algorithm based on the Frank-Wolfe method and AdaBound optimization algorithm to eliminate costly projection steps in large-scale problems

(ii) We prove the convergence of FWAdaBound under the online learning framework. Moreover, we also show that the regret of FWAdaBound is $O(T^{3/4})$, where T is a time horizon

(iii) We present various of experiments to validate computation cost reduction of FWAdaBound and show good generalization ability of FWAdaBound on public dataset

The rest of this paper is organized as follows: in Section 2, we review some important related work of FWAdaBound. In Section 3, we introduce preliminary knowledge about optimization object and online learning. In Sections 4, we present some frequently used assumptions and detailed design of FWAdaBound. In Section 5, we prove the convergence of FWAdaBound in theory and obtain the regret bound. In Section 6, we conduct various experiments in detail on public datasets. Finally, we present the conclusion of this paper in Section 7.

2. Related Work

SGD performs linear iteration of decision variables based on gradient. Therefore, SGD is one of the simplest and easiest implemented algorithms in deep learning. It has good generalization ability if labeled training samples are sufficient. However, the slow convergence rate of SGD always makes it difficult to converge to optima under limited labeled training samples. To speed up convergence rate of SGD, the first-order momentum and the second-order momentum based on the gradient are used in optimization algorithms. More specifically, the fist-order momentum of the gradient is used to retain historical information of gradient, and the second-order momentum of gradient is used to make the step size adaptive. The first algorithm combining these two momentums of gradient is Adam, which obtains a faster convergence rate than SGD [14]. However, Reddi et al. found that the convergence proof of Adam was problematic and proposed an improved variant of Adam, called AMSGrad [15]. Moreover, [17] advocated that it is beneficial to consider more past gradients when designing adaptive learning rates, and thereby, they proposed NosAdam.

Despite Adam, AMSGrad, and NosAdam both improving the convergence rate, however, these three algorithms all have lower generalization ability than SGD under sufficient training samples. For this reason, [18] proposed SWATS to improve generalization performance by switching from Adam to SGD in the later stages of training. Although SWATS improves generalization ability for adaptive momentum algorithms, its switching time is difficult to be accurately controlled. Based on works mentioned above, [16] analyzed that unstable and extreme learning rates may lead to the lack of generalization performance of adaptive methods. Moreover, [16] used a dynamic boundary of the learning rate, where the upper and lower limits can smoothly converge to a constant final step size, respectively. Furthermore, the algorithm proposed is called AdaBound. Therefore, AdaBound currently performs better in terms of convergence speed and generalization ability compared with other algorithms.

Although AdaBound performs well in the convergence rate and generalization ability, projection steps in AdaBound produce numbers of computation cost and make training process prohibit when dealing with large-scale problems. To be specific, the projection operator defined as $\Pi_{F,M}$ can be formed as follows:

$$\Pi_{\mathcal{F},M}(\mathbf{y}) = \arg\min_{\mathbf{x}\in\mathcal{F}} \left\| M^{1/2}(\mathbf{x}-\mathbf{y}) \right\|, \tag{1}$$

where $\mathcal{F}$ is a convex feasible set, $\mathbf{x}$ is a decision variable in the feasible domain, and $\mathbf{y}$ is an unknown variable.

Equation (1) shows the high-order calculation method of a projection operation, which brings a lot of calculation cost to the algorithm. The efficiency of algorithms like AdaBound are highly dependent on time and hardware. Therefore, it is necessary to eliminate projection steps of AdaBound in order to improve its efficiency. However, this much needed algorithm has not yet been proposed. For this reason, we propose a projection-free algorithm based on AdaBound, which uses the Frank-Wolfe method to replace high-order projection steps with one-dimensional linear searches.

3. Preliminaries

In this section, we first introduce some notations for convenience. Throughout this paper, we let a boldtype letter, like $\mathbf{x}$, denote a vector. For operative symbol, we let $\mathbf{x}/\mathbf{y}$ denote the element-wise division, $\mathbf{x}^2$ denotes the element-wise square, and $\sqrt{\mathbf{x}}$ denotes the element-wise square root. For the tth iteration, we let $\mathbf{x}_t$ denote the decision vector, f_t denote the cost function, and $x_{t,i}$ denote the ith coordinate of $\mathbf{x}_t$. Moreover, diag $\{\mathbf{x}\}$ denotes a diagonal matrix generated by the elements of $\mathbf{x}$ in order, max $\{\cdot, \cdot\}$ represents element-wise maximum, and $\langle \cdot, \cdot \rangle$ denotes the scalar inner product. In addition, we let R denote the real number set and $\Pi_{\mathscr{F},M}(\cdot)$ denotes the weighted projection operation, where $\mathscr{F}$ is a feasible set and M represents a positive definite matrix.

In this paper, we consider an online convex optimization problem, in which the cost function changes over the time or iteration. If $\mathscr{F} \subset R$ is a convex and compact set, the decision vector $\mathbf{x} \in \mathscr{F}$, and the cost function at time t is f_t; then, we focus on the following optimization objective:

$$\min_{\mathbf{x}\in\mathscr{F}} \sum_{t=1}^{T} f_t(\mathbf{x}). \tag{2}$$

In order to solve the online optimization problem, i.e., Equation (2), an online optimization algorithm is required. In addition, to measure the performance of an online optimization algorithm, one of standard approaches is regret. Moreover, if we let $\mathbf{x}^*$ denote the theoretical optimal solution, then the definition of regret is as follows:

$$R(T) = \sum_{t=1}^{T} f_t(\mathbf{x}_t) - \sum_{t=1}^{T} f_t(\mathbf{x}^*), \tag{3}$$

where $t = 1, \cdots, T$, and $\mathbf{x}^* = \arg\ \min_{\mathbf{x}\in\mathscr{F}} f(\mathbf{x})$.

4. Algorithm Design and Assumptions

In this section, the proposed algorithm design will be firstly introduced in detail. Then, to analyze the convergence of the proposed algorithm, we present some reasonable assumptions.

4.1. Algorithm Design. The input of FWAdaBound is $\mathbf{x}_1 \in \mathscr{F}$, where $\mathscr{F}$ is a convex and compact set. The parameter $\beta_{1t} \in [0, 1)$ and let $\beta_{11} = \beta_1$. Moreover, the parameter $\beta_2 \in [0, 1)$. In addition, the parameters $\alpha, \eta \in (0, 1]$. Let $\mathbf{g}_t$ denote the gradient at time $t \in \{1, \cdots, T\}$; thus, $\mathbf{g}_t = \nabla f_t(\mathbf{x}_t)$. The overall idea of our algorithm is as follows:

At first, we use the first-order momentum of the gradient $\mathbf{u}_t$ to define the sum function for time t: $S_t(\mathbf{x}) = \eta\langle \sum_{\tau=1}^{t} \mathbf{u}_\tau, \mathbf{x}\rangle + \|\mathbf{x} - \mathbf{x}_1\|^2$; then, we use this function to implement one-dimensional linear search $\mathbf{w}_t = \arg\ \min_{\mathbf{x}\in\mathscr{F}} \langle \nabla S_t(\mathbf{x}_t), \mathbf{x}\rangle$, which can accelerate convergence and avoid projection operators, so it is the key of FWAdaBound to reduce the computational cost. Next, we introduce second-order momentum $\mathbf{d}_t$ and use it to generate the dynamic upper bound of learning rate ϖ_t adaptively. Finally, we apply ϖ_t to update the decision variable as $\mathbf{x}_{t+1} = \mathbf{x}_t + \varpi_t e(\mathbf{w}_t - \mathbf{x}_t)$. The specific algorithm is shown in Algorithm 1.

The first-order momentum of the gradient, $\mathbf{u}_t$, is computed by FWAdaBound for time t as follows:

$$\mathbf{u}_t = \beta_{1t}\mathbf{u}_{t-1} + (1 - \beta_{1t})\mathbf{g}_t. \tag{4}$$

The first-order momentum is generated by weighted average of the current gradient and the historical gradient, which speed up convergence rate for optimization algorithms. Next, to implement one-dimensional linear search which replaces of the projection operators, we define the following sum function for time t:

$$S_t(\mathbf{x}) = \eta\left\langle \sum_{\tau=1}^{t} \mathbf{u}_\tau, \mathbf{x} \right\rangle + \|\mathbf{x} - \mathbf{x}_1\|^2. \tag{5}$$

To reduce the computational cost of the projection operation, FWAdaBound searches the feasible variable, $\mathbf{w}_t$, through one-dimensional linear as follows:

$$\mathbf{w}_t = \arg\ \min_{\mathbf{x}\in\mathscr{F}} \langle \nabla S_t(\mathbf{x}_t), \mathbf{x}\rangle. \tag{6}$$

Moreover, to realize the adaptive learning rate, FWAdaBound computes the second-order momentum of the gradient, $\mathbf{d}_t$, for time t as follows:

$$\mathbf{d}_t = \beta_2\mathbf{d}_{t-1} + (1 - \beta_2)\mathbf{g}_t^2. \tag{7}$$

To ensure the convergence of the proposed algorithm, FWAdaBound chooses the bigger value from $\{\mathbf{d}_t, \mathbf{d}_{t-1}\}$ for time t, i.e. $\widehat{\mathbf{d}}_\mathbf{t} = \max\{\mathbf{d}_t, \mathbf{d}_{t-1}\}$. In addition, the diagonal matrix, D_t, based on $\widehat{\mathbf{d}}_\mathbf{t}$ is defined as $D_t = \text{diag}\{\widehat{\mathbf{d}}_t\}$. Next, FWAdaBound generates a dynamic bound for learning rate at time t:

$$\varpi_t = \text{Clip}\left\{\frac{\alpha_t}{\sqrt{D_t}}, \frac{\varpi_{\text{low}}(t)}{\sqrt{t}}, \frac{\varpi_{\text{upp}}(t)}{\sqrt{t}}\right\}, \tag{8}$$

where $\varpi_{\text{low}}(t)$ is the lower bound and $\varpi_{\text{upp}}(t)$ is the upper bound. Therefore, Equation (5) clips the output of $\alpha_t/\sqrt{D_t}$ between the low bound and the upper bound. Finally, FWAdaBound updates the decision variable for time $t + 1$ as follows:

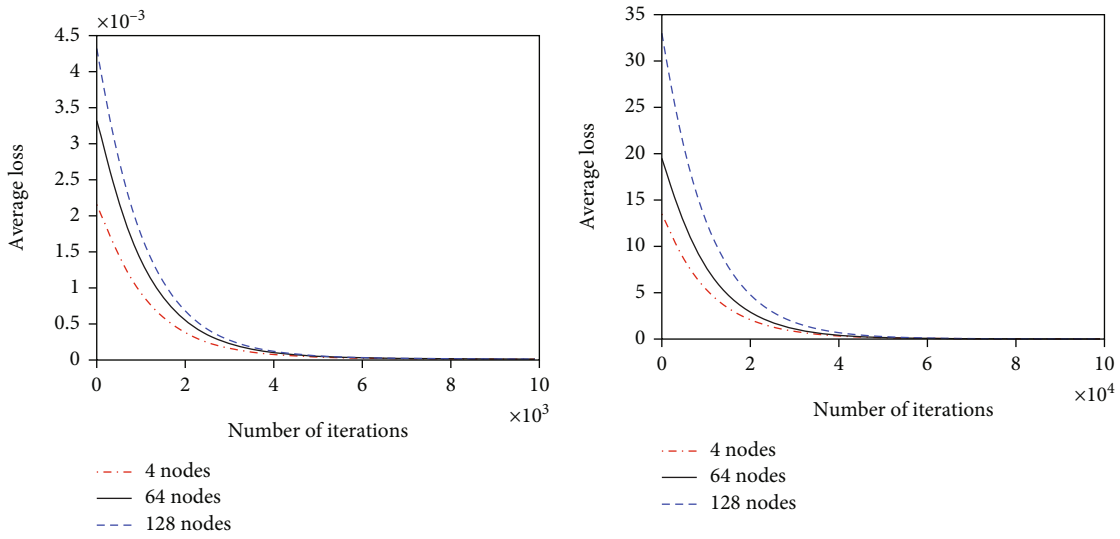
$$\mathbf{x}_{t+1} = \mathbf{x}_t + \varpi_t \bigodot (\mathbf{w}_t - \mathbf{x}_t). \tag{9}$$

Therefore, the design of the proposed algorithm is introduced completely. And we present some following common assumptions, which are the premises of the convergence of the algorithm.

4.2. Assumptions. Next, three assumptions are presented for the proposed algorithm as follows.

Input: $\mathbf{x}_1$
Parameter: $\mathbf{x}_1 \in \mathcal{F}$, and $\beta_{1t} \in [0, 1)$ where $\beta_{11} = \beta_1$, $\beta_2 \in [0, 1)$. Moreover, $\alpha, \eta \in (0, 1]$.
Initially set: $\mathbf{m}_1 = \mathbf{0}$ and $\mathbf{v}_1 = \mathbf{0}$.
Output: $\mathbf{x}_{t+1}$
for $t = 1, 2, 3, \cdots$ **do**
  $t \longleftarrow t + 1$
  Compute gradient of decision variables at time t:
  $\mathbf{g}_t = \nabla f_t(\mathbf{x}_t)$
  Compute the first-order momentum at time t:
  $\mathbf{u}_t = \beta_{1t}\mathbf{u}_{t-1} + (1 - \beta_{1t})\mathbf{g}_t$
  Generate a new sum function:
  $S_t(\mathbf{x}) = \eta\sum_{\tau=1}^{t} \boxtimes \mathbf{u}_\tau, \mathbf{x} + \|\mathbf{x} - \mathbf{x}_1\|^2$
  Search $\mathbf{w}_t$ by one-dimensional linearly:
  $\mathbf{w}_t = \arg\ \min_{\mathbf{x}\in\mathcal{F}} \nabla S_t(\mathbf{x}_t), \mathbf{x}$
  Compute the second-order momentum at time t:
  $\mathbf{d}_t = \beta_2\mathbf{d}_{t-1} + (1 - \beta_2)\mathbf{g}_t^2$
  Select a bigger value for the second-order momentum:
  $\widehat{\mathbf{d}}_t = \max\{\mathbf{d}_t, \mathbf{d}_{t-1}\}$ and $D_t = \text{diag}\{\widehat{\mathbf{d}}_t\}$
  Compute the dynamic bound for learning rate at time t:
  $\varpi_t = \text{Clip}\{(\alpha_t/\sqrt{D_t}), (\varpi_{\text{low}}(t)/\sqrt{t}), (\varpi_{\text{upp}}(t)/\sqrt{t})\}$
  Update the decision variables for time $t + 1$:
  $\mathbf{x}_{t+1} = \mathbf{x}_t + \varpi_t \mathbf{e}(\mathbf{w}_t - \mathbf{x}_t)$
end for
return $\mathbf{x}_{t+1}$

Algorithm 1: FWAdaBound



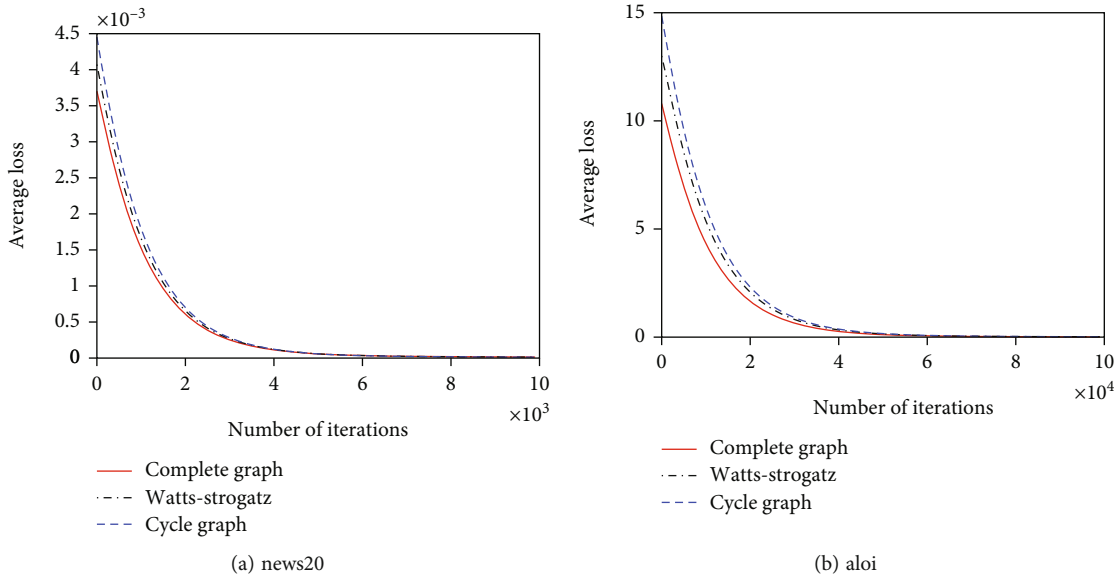
(a) news20 (b) aloi

Figure 1: Comparison of the relationship between the average loss and the running time of each algorithm.

Assumption 1. The constraint set $\mathcal{F}$ is convex and compact. Moreover, the set $\mathcal{F}$ is bounded, i.e., $\|\mathbf{x} - \mathbf{y}\|_\infty \le B_\infty$ for all $\mathbf{x}, \mathbf{y} \in \mathcal{F}$, where $B_\infty > 0$.

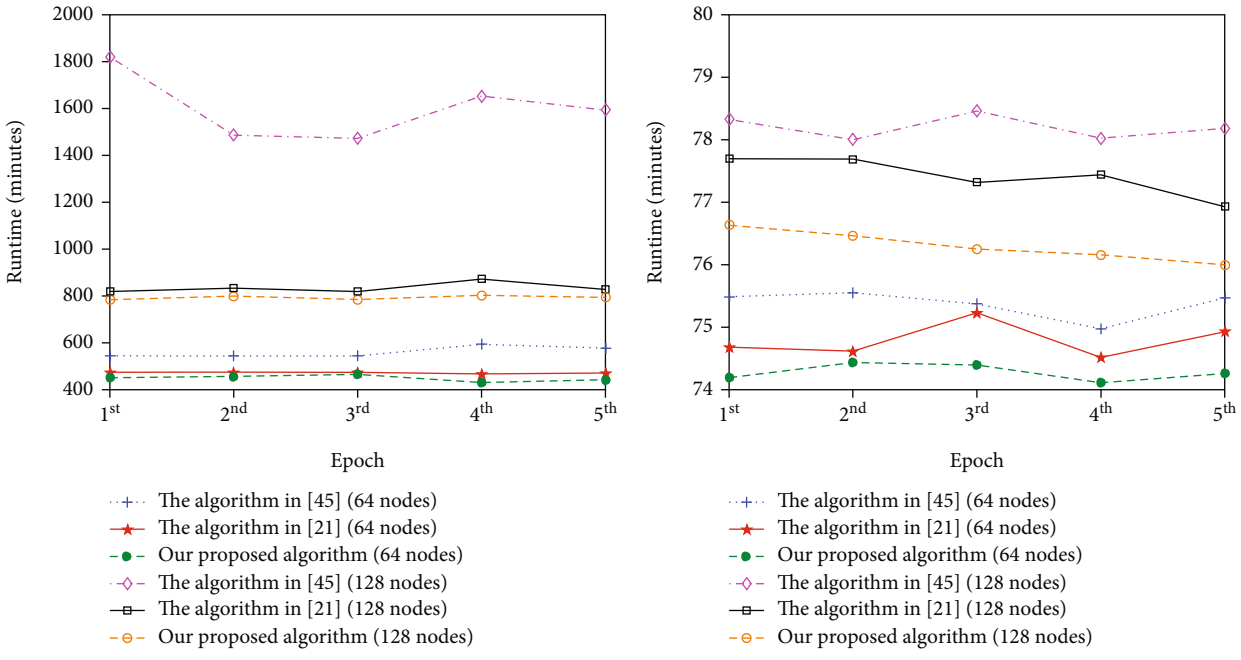
Assumption 2. The cost function f_t of FWAdaBound is convex and differentiable on $\mathcal{F}$ for all $t \in \{1, \cdots, T\}$. In addition, all the cost functions, $\{f_1, \cdots, f_t\}$, are Lipschitz functions with L constant, where $L > 0$.

Assumption 3. The gradient of decision variable $\mathbf{x}_t$ is bounded for all $t \in \{1, 2, \cdots, T\}$ over $\mathcal{F}$, i.e., $\mathbf{g}_t = \|\nabla f_t(\mathbf{x}_t)\| \le G_\infty$, where $G_\infty > 0$.



(a) news20 (b) aloi

Figure 2: Comparison of the relationship between the training accuracy and the running time of each algorithm.



(a) news20 (b) aloi

Figure 3: Comparison of the relationship between the test accuracy and the running time of each algorithm.

Assumption 1 is one of the most basic assumptions of projection-free method, and almost all projection-free related articles use it, such as these classic articles [19, 20]. Assumption 2–3 are supposed commonly and reasonably for analyzing convergence of optimization algorithms such as these research works [14–16]. Next, we present the convergence analysis of the proposed algorithms based on Assumptions 1–3.

In many research works [14–16], Assumptions 1–3 were supposed commonly and reasonably for analyzing convergence of proposed algorithms. Next, we present the convergence analysis of the proposed algorithms based on Assumptions 1–3.

5. Convergence Analysis

We first introduce the following definitions as the beginning of this section. Moreover, the introduced definitions are standard and common in convex optimization.

$$|f(\mathbf{x}) - f(\mathbf{y})| \leq L\|\mathbf{x} - \mathbf{y}\|, \tag{10}$$

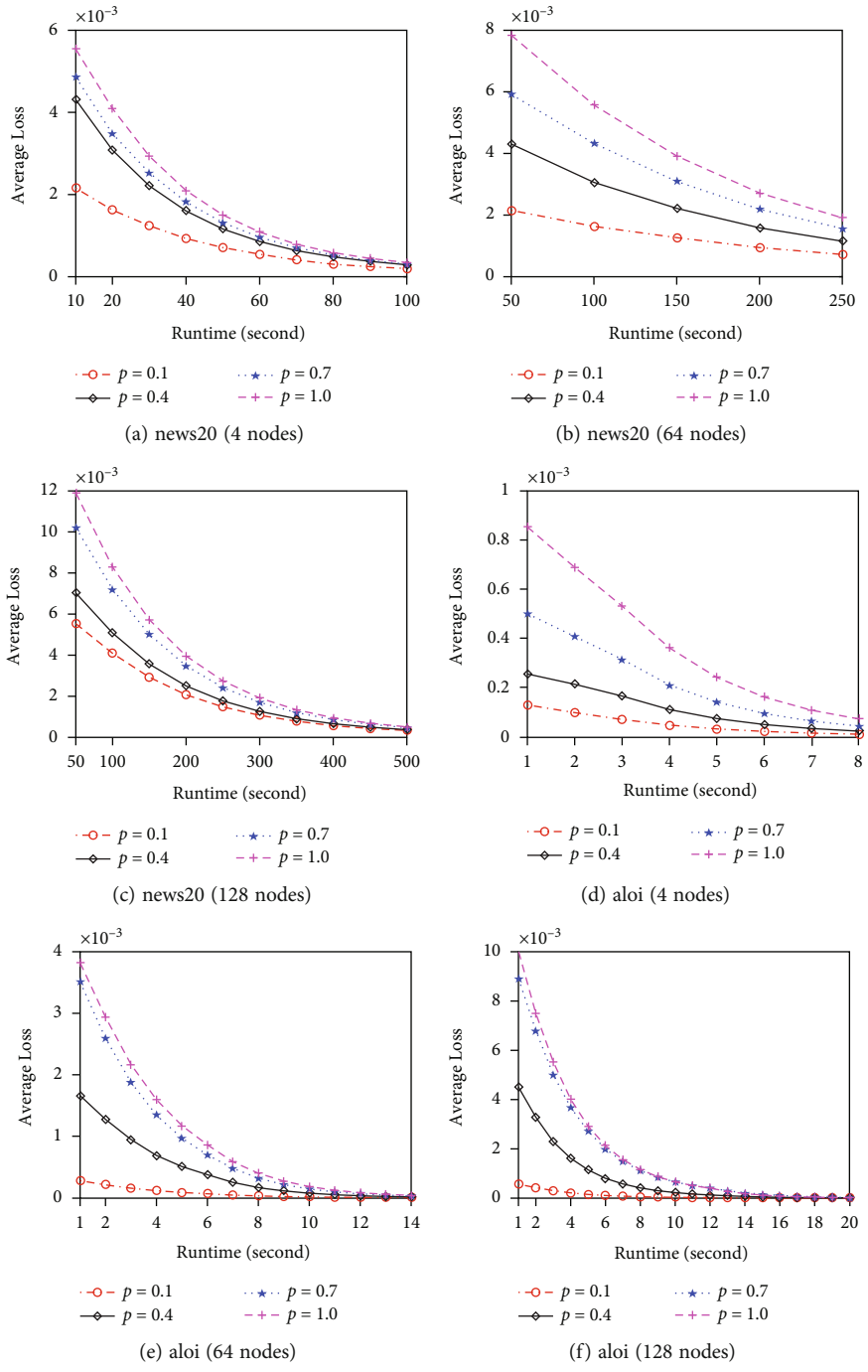


Figure 4: Comparison of the relationship between the perplexity and the running time of each algorithm. Lower is better.

Definition 4. A function $f : \mathcal{F} \mapsto R$ is called L-Lipchitz if for any two points $\mathbf{x}, \mathbf{y} \in \mathcal{F}$ we have

where L is a positive constant.

Definition 5. A function $f : \mathcal{F} \mapsto R$ is convex and differentiable if for all $\mathbf{x}, \mathbf{y} \in \mathcal{F}$, we have

$$f(\mathbf{x}) - f(\mathbf{y}) \leq \nabla f(\mathbf{x})^T (\mathbf{x} - \mathbf{y}). \tag{11}$$

Definition 6. Let $f : \mathcal{F} \mapsto R$ be an arbitrary convex function. Then, the function f is also called μ-smooth if for any two points $\mathbf{x}, \mathbf{y} \in \mathcal{F}$, we have

$$f(\mathbf{x}) - f(\mathbf{y}) \geq \nabla f(\mathbf{x})^T (\mathbf{x} - \mathbf{y}) - \frac{\mu}{2} \|\mathbf{y} - \mathbf{x}\|^2, \tag{12}$$

where $\mu > 0$.

Definition 7. Let $f : \mathcal{F} \mapsto R$ be an arbitrary convex function. Then, the function f is δ-strongly convex if for all $\mathbf{x}, \mathbf{y} \in \mathcal{F}$,

we have

$$f(\mathbf{x}) - f(\mathbf{y}) \leq \nabla f(\mathbf{x})^T(\mathbf{x}-\mathbf{y}) - \frac{\delta}{2}\|\mathbf{y}-\mathbf{x}\|^2, \tag{13}$$

where $\delta > 0$.

In addition, if a function f is δ-strongly convex and let $\mathbf{x}^* = \arg\ \min_{\mathbf{x}\in\mathcal{F}} f(\mathbf{x})$, then we have

$$f(\mathbf{x}) - f(\mathbf{x}^*) \geq \frac{\delta}{2}\|\mathbf{x}-\mathbf{x}^*\|^2. \tag{14}$$

Moreover, from Definition 6, we obtain the following equivalent relation:

$$\|\nabla f(\mathbf{x}) - \nabla f(\mathbf{y})\| \leq \mu\|\mathbf{x}-\mathbf{y}\|, \tag{15}$$

where $\mathbf{x}, \mathbf{y} \in \mathcal{F}$.

In order to simplify the process of convergence analysis, we define some intermediate variables. Let $\mathbf{x}_t^* = \arg\ \min_{\mathbf{x}\in\mathcal{F}} S_t(\mathbf{x})$ for any $t \in \{1, \cdots, T\}$. Moreover, we define $S_0(\mathbf{x}) = \|\mathbf{x} - \mathbf{x}_1\|^2$ at time $t = 0$. In addition, we present the following relation for S_t:

$$z_t(\mathbf{x}) = S_t(\mathbf{x}) - S_t(\mathbf{x}_t^*). \tag{16}$$

When $\mathbf{x} = \mathbf{x}_t$, for brevity, denoting $z_t = z_t(\mathbf{x}_t)$. Next, we present the following Lemma 8 to bound z_{t+1}.

Lemma 8. *If Assumptions 1–3 are satisfied, and variables* $\{\mathbf{x}_t\}, \{\mathbf{u}_t\}$, *and* $\{\mathbf{d}_t\}$ *are generated by Algorithm 1 for* $t \in \{1, \cdots, T\}$, *and* $\beta_{1t} = \beta_1\lambda^{t-1} \leq \beta_{1(t-1)} \leq \beta_1$, *where* $\lambda \in (0, 1]$. *In addition, suppose that* $0 \leq \varpi_{low}(t) \leq \varpi_{low}(t+1)$, *and* $0 \leq \varpi_{upp}(t+1) \leq \varpi_{upp}(t)$, *denoting* $B_\infty = \varpi_{upp}(1)$ *and* $C_\infty = \varpi_{low}(1)$. *Then, we have*

$$z_{t+1} \leq \left(1 - \frac{B_\infty^2}{\sqrt{t}}\right) z_t + \frac{n\eta G_\infty}{1-\beta_1}\sqrt{z_{t+1}} + \frac{B_\infty^4}{t}. \tag{17}$$

Proof. By Definition 6, we know that $S_t(\mathbf{x})$ is a 2-smooth function. Moreover, using the definition of $z_t(\mathbf{x})$ (i.e., Equation (16)) and $\mathbf{x}_t$, we obtain the following relation:

$$z_t(\mathbf{x}_{t+1}) = S_t(\mathbf{x}_{t+1}) - S_t(\mathbf{x}_t^*) = S_t(\mathbf{x}_t + \varpi_t \mathbf{e}(\mathbf{w}_t - \mathbf{x}_t)) - S_t(\mathbf{x}_t^*). \tag{18}$$

□

From the bounds of $\varpi_{\text{low}}(t)$ and $\varpi_{\text{upp}}(t)$, we have the following:

$$z_t(\mathbf{x}_{t+1}) \leq S_t\left(\mathbf{x}_t + \frac{\varpi_{\text{upp}}(1)}{\sqrt{t}}\mathbf{e}(\mathbf{w}_t - \mathbf{x}_t)\right) - S_t(\mathbf{x}_t^*) \leq S_t\left(\mathbf{x}_t + \frac{B_\infty}{\sqrt{t}}\mathbf{e}(\mathbf{w}_t - \mathbf{x}_t)\right) - S_t(\mathbf{x}_t^*). \tag{19}$$

In addition, from the Definition 7 and the strong-convexity of $S_t(\mathbf{x})$, we obtain the following:

$$z_t(\mathbf{x}_{t+1}) \leq S_t(\mathbf{x}_t) - S_t(\mathbf{x}_t^*) + \frac{B_\infty^2}{t}\|\mathbf{w}_t - \mathbf{x}_t\|^2 + \frac{B_\infty^2}{\sqrt{t}}\langle \nabla S_t(\mathbf{x}_t), (\mathbf{w}_t - \mathbf{x}_t)\rangle. \tag{20}$$

From the definition of $\mathbf{w}_t$, we attain the following relation:

$$\langle \nabla S_t(\mathbf{x}_t), \mathbf{w}_t\rangle \leq \langle \nabla S_t(\mathbf{x}_t), \mathbf{x}_t^*\rangle. \tag{21}$$

Moreover, from Equation (21), we have the following:

$$\langle \nabla S_t(\mathbf{x}_t), (\mathbf{w}_t - \mathbf{x}_t)\rangle \leq \langle \nabla S_t(\mathbf{x}_t), (\mathbf{x}_t^* - \mathbf{x}_t)\rangle. \tag{22}$$

Plugging Equation (22) into Equation (20), we attain the following relation:

$$z_t(\mathbf{x}_{t+1}) \leq S_t(\mathbf{x}_t) - S_t(\mathbf{x}_t^*) + \frac{B_\infty^2}{t}\|\mathbf{w}_t - \mathbf{x}_t\|^2 + \frac{B_\infty^2}{\sqrt{t}}\langle \nabla S_t(\mathbf{x}_t), (\mathbf{x}_t^* - \mathbf{x}_t)\rangle. \tag{23}$$

According to Definition 5 and the convexity of $S_t(\mathbf{x})$, we obtain the following relation:

$$\langle \nabla S_t(\mathbf{x}_t), (\mathbf{x}_t^* - \mathbf{x}_t)\rangle \leq S_t(\mathbf{x}^*) - S_t(\mathbf{x}_t). \tag{24}$$

Furthermore, plugging Equation (24) into Equation (23), we get the following relation:

$$\begin{aligned} z_t(\mathbf{x}_{t+1}) &\leq S_t(\mathbf{x}_t) - S_t(\mathbf{x}_t^*) + \frac{B_\infty^2}{t}\|\mathbf{w}_t - \mathbf{x}_t\|^2 + \frac{B_\infty^2}{\sqrt{t}}(S_t(\mathbf{x}^*) - S_t(\mathbf{x}_t)) \\ &\leq \left(1 - \frac{B_\infty^2}{\sqrt{t}}\right)(S_t(\mathbf{x}_t) - S_t(\mathbf{x}_t^*)) + \frac{B_\infty^2}{t}\|\mathbf{w}_t - \mathbf{x}_t\|^2. \end{aligned} \tag{25}$$

Next, we consider the term $z_t(\mathbf{x}_{t+1})$ in Equation (25). By the definition of $z_t(\mathbf{x})$, we first obtain the following relation:

$$z_{t+1}(\mathbf{x}_{t+1}) \leq S_{t+1}(\mathbf{x}_{t+1}) - S_{t+1}(\mathbf{x}_{t+1}^*). \tag{26}$$

Then, transforming Equation (26), and we attain the following relation:

$$z_{t+1}(\mathbf{x}_{t+1}) \leq S_t(\mathbf{x}_{t+1}) - S_t(\mathbf{x}_{t+1}^*) + S_{t+1}(\mathbf{x}_{t+1}) - S_t(\mathbf{x}_{t+1}) + S_{t+1}(\mathbf{x}_{t+1}^*) - S_t(\mathbf{x}_{t+1}^*). \tag{27}$$

In addition, due to the fact that $\mathbf{x}_t^* = \arg\ \min_{\mathbf{x}\in\mathcal{F}} S_t(\mathbf{x})$, we have $S_t(\mathbf{x}_t^*) \leq S_t(\mathbf{x}_{t+1}^*)$. For this reason, we obtain the following relation from Equation (27):

$$\begin{aligned} z_{t+1}(\mathbf{x}_{t+1}) &\leq S_t(\mathbf{x}_{t+1}) - S_t(\mathbf{x}_t^*) + S_{t+1}(\mathbf{x}_{t+1}) - S_t(\mathbf{x}_{t+1}) + S_{t+1}(\mathbf{x}_{t+1}^*) - S_t(\mathbf{x}_{t+1}^*) \\ &= z_t(\mathbf{x}_{t+1}) + S_{t+1}(\mathbf{x}_{t+1}) - S_t(\mathbf{x}_{t+1}) + S_{t+1}(\mathbf{x}_{t+1}^*) - S_t(\mathbf{x}_{t+1}^*). \end{aligned} \tag{28}$$

Next, we consider the terms $S_{t+1}(\mathbf{x}_{t+1}) - S_t(\mathbf{x}_{t+1})$ and $S_{t+1}(\mathbf{x}_{t+1}^*) - S_t(\mathbf{x}_{t+1}^*)$ in Equation (28). From the definition

of $S_t(\mathbf{x})$, we have the following relation:

$$S_t(\mathbf{x}) - S_t(\mathbf{x}) = \eta \sum_{\tau=1}^{t+1} \mathbf{u}_\tau^T \mathbf{x} + \|\mathbf{x} - \mathbf{x}_1\|^2 - \eta \sum_{\tau=1}^{t} \mathbf{u}_\tau^T \mathbf{x} - \|\mathbf{x} - \mathbf{x}_1\|^2 = \eta \mathbf{u}_{t+1}^T \mathbf{x}. \tag{29}$$

Let $\mathbf{x} = \mathbf{x}_{t+1}$ in Equation (29), we obtain

$$S_t(\mathbf{x}_{t+1}) - S_t(\mathbf{x}_{t+1}) = \eta \mathbf{u}_{t+1}^T \mathbf{x}_{t+1}. \tag{30}$$

In addition, let $\mathbf{x} = \mathbf{x}_{t+1}^*$ in Equationss (29), we attain

$$S_t(\mathbf{x}_{t+1}^*) - S_t(\mathbf{x}_{t+1}^*) = \eta \mathbf{u}_{t+1}^T \mathbf{x}_{t+1}^*. \tag{31}$$

Combining Equations (28), (30), and (31), we have the following:

$$\begin{aligned} z_{t+1}(\mathbf{x}_{t+1}) &\le z_t(\mathbf{x}_{t+1}) + \eta \mathbf{u}_{t+1}^T \mathbf{x}_{t+1} + \eta \mathbf{u}_{t+1}^T \mathbf{x}_{t+1}^* = z_t(\mathbf{x}_{t+1}) + \eta \mathbf{u}_{t+1}^T (\mathbf{x}_{t+1} - \mathbf{x}_{t+1}^*) \\ &\le z_t(\mathbf{x}_{t+1}) + \eta \left\| \mathbf{u}_{t+1}^T \right\| \|\mathbf{x}_{t+1} - \mathbf{x}_{t+1}^*\|. \end{aligned} \tag{32}$$

The second inequality in Equation (32) follows from Cauchy-Schwarz inequality. Besides, plugging Equation (23) into Equation (32), we obtain

$$z_{t+1}(\mathbf{x}_{t+1}) \le \left(1 - \frac{B_\infty^2}{\sqrt{t}}\right)(S_t(\mathbf{x}_t) - S_t(\mathbf{x}_t^*)) + \frac{B_\infty^2}{t}\|\mathbf{w}_t - \mathbf{x}_t\|^2 + \eta \left\| \mathbf{u}_{t+1}^T \right\| \|\mathbf{x}_{t+1} - \mathbf{x}_{t+1}^*\|. \tag{33}$$

Since $z_t(\mathbf{x}_t) = S_t(\mathbf{x}_t) - S_t(\mathbf{x}_t^*)$, we attain the following relation from Equation ((33))

$$\begin{aligned} z_{t+1}(\mathbf{x}_{t+1}) &\le \left(1 - \frac{B_\infty^2}{\sqrt{t}}\right) z_t(\mathbf{x}_t) + \frac{B_\infty^2}{t}\|\mathbf{w}_t - \mathbf{x}_t\|^2 + \eta \left\| \mathbf{u}_{t+1}^T \right\| \|\mathbf{x}_{t+1} - \mathbf{x}_{t+1}^*\| \\ &\le \left(1 - \frac{B_\infty^2}{\sqrt{t}}\right) z_t + \frac{B_\infty^2}{t}\|\mathbf{w}_t - \mathbf{x}_t\|^2 + \eta \left\| \mathbf{u}_{t+1}^T \right\| \|\mathbf{x}_{t+1} - \mathbf{x}_{t+1}^*\|. \end{aligned} \tag{34}$$

The second inequality in Equation (34) follows the definition of $z_t = z_t(\mathbf{x}_t)$.

Before estimating the bound of $z_{t+1}(\mathbf{x}_{t+1})$, we consider the bound of $\|\mathbf{u}_{t+1}\|$. To this end, applying the recursive algorithm on $\mathbf{u}_t$, we have

$$\mathbf{u}_t = \sum_{j=1}^{t} \left(1 - \beta_{1j}\right) \prod_{k=1}^{t-j} \beta_{1(t-k+1)} \mathbf{g}_j. \tag{35}$$

From Assumption 3 and Equation (35), we attain

$$\begin{aligned} \|\mathbf{u}_t\| &\le \sum_{j=1}^{t} \left(1 - \beta_{1j}\right) \prod_{k=1}^{t-j} \beta_{1(t-k+1)} \left(\sum_{\sigma=1}^{n} g_{j,\sigma} \right) \\ &\le nG_\infty \sum_{j=1}^{t} \left(1 - \beta_{1j}\right) \prod_{k=1}^{t-j} \beta_{1(t-k+1)} \le nG_\infty \sum_{j=1}^{t} \prod_{k=1}^{t-j} \beta_{1(t-k+1)}. \end{aligned} \tag{36}$$

In addition, because $\beta_{1t} = \beta_1 \lambda^{t-1}$, we obtain the following relation from Equation (36):

$$\|\mathbf{u}_t\| \le nG_\infty \sum_{j=1}^{t} \beta_1^{t-j} \le \frac{nG_\infty}{1 - \beta_1}. \tag{37}$$

Plugging Equation (37) into Equation (34), we attain

$$z_{t+1} \le \left(1 - \frac{B_\infty^2}{\sqrt{t}}\right) z_t + \frac{B_\infty^2}{t}\|\mathbf{w}_t - \mathbf{x}_t\|^2 + \frac{n\eta G_\infty}{1 - \beta_1}\|\mathbf{x}_{t+1} - \mathbf{x}_{t+1}^*\|. \tag{38}$$

Since $\mathbf{w}_t = \arg\min_{\mathbf{x} \in \mathscr{F}} \langle \nabla S_t(\mathbf{x}_t), \mathbf{x} \rangle$, we have $\mathbf{w}_t \in \mathscr{F}$. Moreover, from Assumption 1, we have

$$\|\mathbf{w}_t - \mathbf{x}_t\| \le B_\infty. \tag{39}$$

Hence, combining Equations (38) and (40), we obtain

$$z_{t+1} \le \left(1 - \frac{B_\infty^2}{\sqrt{t}}\right) z_t + \frac{n\eta G_\infty}{1 - \beta_1}\|\mathbf{x}_{t+1} - \mathbf{x}_{t+1}^*\| + \frac{B_\infty^4}{t}. \tag{40}$$

Applying $\delta = 2$ on Definition 7, we attain that the function $S_t(\mathbf{x})$ is 2-strongly convex. In addition, from Equation (14), we have

$$\|\mathbf{x}_{t+1} - \mathbf{x}_{t+1}^*\| \le \sqrt{S_{t+1}(\mathbf{x}_{t+1}) - S_{t+1}(\mathbf{x}_{t+1}^*)} = \sqrt{z_{t+1}}. \tag{41}$$

Moreover, substituting Equation (41) into Equation (40), and we can obtain the following:

$$z_{t+1} \le \left(1 - \frac{B_\infty^2}{\sqrt{t}}\right) z_t + \frac{n\eta G_\infty}{1 - \beta_1}\sqrt{z_{t+1}} + \frac{B_\infty^4}{t}. \tag{42}$$

Therefore, the proof of Lemma 8 is completed.

Now, we get the iterative relations between z_t and z_{t+1} from Lemma 8. In order to attain the final bound of z_t, we should present the following two lemmas. First of all, we introduce the first lemma of the two lemmas.

Lemma 9. *For all $t = 1, 2, \cdots$, we can obtain the following relation:*

$$\frac{1}{\sqrt{t}}\left(1 - \frac{1}{2\sqrt{t}}\right) \le \frac{1}{\sqrt{t+1}}. \tag{43}$$

Proof. We first square both sides of Equation (43) and take

the difference; then, we have the following relation:

$$\begin{aligned}\left(\frac{1}{\sqrt{t}}\left(1-\frac{1}{2\sqrt{t}}\right)\right)^2-\left(\frac{1}{\sqrt{t+1}}\right)^2&=\frac{1}{t}-\frac{1}{t\sqrt{t}}+\frac{1}{4t^2}-\frac{1}{t+1}\\&=\left(\frac{1}{t}-\frac{1}{t\sqrt{t}}+\frac{1}{4t^2}-\frac{1}{t+1}\right)\frac{t(t+1)}{t(t+1)}\\&=\frac{t+1+(t+1/4t)-t+1/\sqrt{t}-t}{t(t+1)}\\&=\frac{t+(t+1/4)-(t+1)\sqrt{t}}{t^2(t+1)}=\frac{(5t+1)-4\sqrt{t}(t+1)\sqrt{t}}{t^2(t+1)}.\end{aligned}\tag{44}$$

□

Observing terms $(5t+1)$ and $4\sqrt{t}(t+1)\sqrt{t}$, we can know that they all increase with t, and the latter grows faster than the former. Therefore, we can further attain the following relation from Equation (44):

$$\left(\frac{1}{\sqrt{t}}\left(1-\frac{1}{2\sqrt{t}}\right)\right)^2-\left(\frac{1}{\sqrt{t+1}}\right)^2=\frac{(5t+1)-4\sqrt{t}(t+1)\sqrt{t}}{t^2(t+1)}\le 0.\tag{45}$$

In addition, combining Equations (44) and (43), we obtain the following relation:

$$\left(\frac{1}{\sqrt{t}}\left(1-\frac{1}{2\sqrt{t}}\right)\right)^2\le\left(\frac{1}{\sqrt{t+1}}\right)^2.\tag{46}$$

From Equation (46), we have the following relation:

$$\frac{1}{\sqrt{t}}\left(1-\frac{1}{2\sqrt{t}}\right)\le\frac{1}{\sqrt{t+1}}.\tag{47}$$

Therefore, the proof of Lemma 9 is completed.

Next, we introduce the last lemma about the final bound of z_t as follows.

Lemma 10. *If Assumptions 1–3 are satisfied, and variables $\{\mathbf{x}_t\},\{\mathbf{u}_t\},\{\mathbf{d}_t\}$ are generated by Algorithm 4.1 for $t=\{1,\cdots,T\}$, and $\beta_{1t}=\beta_1\lambda^{t-1}\le\beta_{1(t-1)}\le\beta_1$, where $\lambda\in(0,1]$. In addition, suppose that $0\le\varpi_{low}(t)\le\varpi_{low}(t+1)$, and $0\le\varpi_{upp}(t+1)\le\varpi_{upp}(t)$. Denoting $B_\infty=\varpi_{upp}(1)$, and $C_\infty=\varpi_{low}(1)$. Moreover, as the parameters n, η, and β_1 are chosen such that $(n\eta G_\infty/(1-\beta_1)1-\beta_1)\sqrt{z_{t+1}}\le(3B_\infty^4-2B_\infty^2)/t$ by Algorithm 1, we then have*

$$z_{t+1}\le\frac{4B_\infty^2}{\sqrt{t+1}}.\tag{48}$$

Proof. By Equation (42), we have the following relation:

$$z_{t+1}\le\left(1-\frac{B_\infty^2}{\sqrt{t}}\right)z_t+\frac{4B_\infty^4-2B_\infty^2}{t}.\tag{49}$$

□

Now, we can use mathematical induction to get the bound of z_t. First, when $t=1$, from definition of z_t, we attain

$$z_1\le S_1(\mathbf{x}_1)-S_1(\mathbf{x}_1^*)\le\|\mathbf{x}_1-\mathbf{x}_1\|^2-\|\mathbf{x}_1^*-\mathbf{x}_1\|^2=-\|\mathbf{x}_1^*-\mathbf{x}_1\|^2\le 4B_\infty^2.\tag{50}$$

Therefore, the base of mathematical induction is true for $t=1$. Second, supposing that the mathematical induction is also true for t, and we present that it also true for $t+1$ as follows:

$$\begin{aligned}z_{t+1}&\le\left(1-\frac{B_\infty^2}{\sqrt{t}}\right)\frac{4B_\infty^2}{\sqrt{t}}+\frac{4B_\infty^4-2B_\infty^2}{t}\le\frac{4B_\infty^2}{\sqrt{t}}-\frac{4B_\infty^4}{t}+\frac{4B_\infty^4}{t}-\frac{2B_\infty^2}{t}\\&\le\frac{4B_\infty^2}{\sqrt{t}}\left(1-\frac{1}{2\sqrt{t}}\right).\end{aligned}\tag{51}$$

Applying Lemma 9 into Equation (51), we obtain

$$z_{t+1}\le\frac{4B_\infty^2}{\sqrt{t+1}}.\tag{52}$$

Therefore, the proof of Lemma 10 is completed.

Next, we present the following result to attain the bound of $R(T)$.

Theorem 11. *If the Assumptions 1–3 are satisfied. Moreover, the sequences $\mathbf{u}_t$, $\mathbf{w}_t$, and $\mathbf{d}_t$ are all generated by our proposed algorithm, which $t\in\{1,2,\cdots,T\}$. Then, we obtain that*

$$R(T)\le\frac{8LB_\infty}{3}T^{3/4}+\frac{B_\infty^2}{\eta(1-\beta_1)\beta_{1T}}.\tag{53}$$

Proof. From Lemma 10 and Equation (41), we have the following:

$$\|\mathbf{x}_t-\mathbf{x}_t^*\|\le\sqrt{z_t}\le\sqrt{\frac{4B_\infty^2}{\sqrt{t}}}=2B_\infty t^{-1/4}.\tag{54}$$

□

Summing over Equation (54) for $t=1,\cdots,T$, we attain the following relation:

$$\sum_{t=1}^{T}\|\mathbf{x}_t-\mathbf{x}_t^*\|\le\sum_{t=1}^{T}2B_\infty t^{-1/4}\le\frac{8B_\infty}{3}T^{3/4}.\tag{55}$$

By Assumption 2, we know that f_t is a Lipschitz function. In addition, applying Definition 4, we have

$$|f_t(\mathbf{x}_t)-f_t(\mathbf{x}_t^*)|\le L\|\mathbf{x}_t-\mathbf{x}_t^*\|.\tag{56}$$

In addition, combining Equation (55) and (56), we obtain

$$\sum_{t=1}^{T}|f_t(\mathbf{x}_t)-f_t(\mathbf{x}_t^*)|\le\sum_{t=1}^{T}L\|\mathbf{x}_t-\mathbf{x}_t^*\|\le\frac{8LB_\infty}{3}T^{3/4}.\tag{57}$$

Moreover, from the definition of $R(T)$, we have

$$R(T) = \sum_{t=1}^{T}[f_t(\mathbf{x}_t) - f_t(\mathbf{x}^*)] = \sum_{t=1}^{T}[f_t(\mathbf{x}_t) - f_t(\mathbf{x}_t^*)] + \sum_{t=1}^{T}[f_t(\mathbf{x}_t^*) - f_t(\mathbf{x}^*)]. \tag{58}$$

Since $S_0(\mathbf{x}) = \|\mathbf{x} - \mathbf{x}_1\|^2$, we attain the following relation:

$$\begin{aligned} S_T(\mathbf{x}_t^*) - S_T(\mathbf{x}^*) &\le \left[\eta\sum_{t=1}^{T}\mathbf{u}_t\mathbf{x}_t^* + \|\mathbf{x}_t^* - \mathbf{x}_1\|^2\right] - \left[\eta\sum_{t=1}^{T}\mathbf{u}_t\mathbf{x}_t^* + \|\mathbf{x}^* - \mathbf{x}_1\|^2\right] \\ &\le \eta\sum_{t=1}^{T}\mathbf{u}_t(\mathbf{x}_t^* - \mathbf{x}^*) + \|\mathbf{x}_t^* - \mathbf{x}_1\|^2 - \|\mathbf{x}^* - \mathbf{x}_1\|^2 \le 0. \end{aligned} \tag{59}$$

Since $\mathbf{x}^*, \mathbf{x}_1 \in \mathcal{F}$, and by Assumption 1, we have $\|\mathbf{x}^* - \mathbf{x}_1\| \le B_\infty$. Moreover, according to Equation (59), we obtain the following relation:

$$\sum_{t=1}^{T}\mathbf{u}_t(\mathbf{x}_t^* - \mathbf{x}^*) \le \frac{1}{\eta}\left[\|\mathbf{x}^* - \mathbf{x}_1\|^2 - \|\mathbf{x}_t^* - \mathbf{x}_1\|^2\right] \le \frac{B_\infty^2}{\eta}. \tag{60}$$

Then, combining Equations (35) and (60), we have the following relation:

$$\begin{aligned} \sum_{t=1}^{T}\mathbf{u}_t(\mathbf{x}_t^* - \mathbf{x}^*) &= \sum_{t=1}^{T}\left[\sum_{j=1}^{t}\left(1-\beta_{1j}\right)\prod_{k=1}^{t-j}\beta_{1(t-k+1)}\mathbf{g}_j\right] \times (\mathbf{x}_t^* - \mathbf{x}^*) \\ &\ge \sum_{t=1}^{T}\left[\sum_{j=1}^{t}\left(1-\beta_{1j}\right)\beta_{1t}\mathbf{g}_j\right](\mathbf{x}_t^* - \mathbf{x}^*) \\ &\ge (1-\beta_1)\beta_{1T}\sum_{t=1}^{T}\mathbf{g}_t(\mathbf{x}_t^* - \mathbf{x}^*). \end{aligned} \tag{61}$$

Therefore, from Equations (60) and (61), we attain the following relation:

$$\sum_{t=1}^{T}\mathbf{g}_t(\mathbf{x}_t^* - \mathbf{x}^*) \le \frac{B_\infty^2}{\eta(1-\beta_1)\beta_{1T}}. \tag{62}$$

Applying Definition 5 and Equation (58), we obtain

$$R(T) \le \sum_{t=1}^{T}|f_t(\mathbf{x}_t) - f_t(\mathbf{x}_t^*)| + \sum_{t=1}^{T}\mathbf{g}_t(\mathbf{x}_t^* - \mathbf{x}^*). \tag{63}$$

Finally, substituting Equations (57) and (62) into Equation (63), we have

$$R(T) \le \frac{8LB_\infty}{3}T^{3/4} + \frac{B_\infty^2}{\eta(1-\beta_1)\beta_{1T}}. \tag{64}$$

Therefore, the proof of Theorem 11 is completed.

From Theorem 11, we get that $\lim_{T\longrightarrow\infty}R(T)/T = 0$, which indicates that our proposed algorithm is convergent. In addition, by Equation (64), we know that the regret bound of our proposed algorithm is $O(T^{3/4})$.

Remark 12. This work is closed to the previous works [14–16]. Our regret bound $O(T^{3/4})$ is worse than the regret bound $O(\sqrt{T})$, which is obtained by [15, 16], but the number of iterations is increased within the same amount of time due to the lower computational cost per iteration. Hence, the overall convergent rate of the proposed algorithm is faster than the Adam-type algorithms such as Adam [14] and AdaBound [16].

Next, we will validate the performance of our proposed algorithm by simulation experiments.

6. Experiments

Our algorithm is mainly used in the field of multimedia communication. Specifically, through modeling and analysis, it is proved that the algorithm can reduce the calculation and storage cost of text, voice, picture, and other information in multimedia information transmission.

So, in this section, we, respectively, apply and compare the proposed algorithm on different dataset to validate the convergence and performance of the proposed algorithm. One application is the image classification on CIFAR-10 [21] dataset, and another one is the language modeling on the Penn Treebank dataset [22]. The experiments are executed on the equipment with 1080Ti GPU and CUDA 0.4.0, and written in Python 3.7 with Torch 1.0.1 framework. The details of experiment settings and results are described in the content below.

6.1. Experiment Settings. The CIFAR-10 is a famous and standard dataset for image classification, which consists of 60,000 32×32 color images in 10 classes, and with 6,000 images per class. Moreover, the dataset has 50,000 training images and 10,000 test images, respectively. We use deep models, ResNet-34 and DenseNet-121, to finish the classification tasks on CIFAR-10. The model ResNet-34, a popular model in deep learning, is a deep residual networks with 34 layers. In addition, DenseNet-121 is a dense convolutional network with 121 layers, in which each layer accepts all preceding layers as its additional input.

The Penn Treebank is a popular and classical dataset for language modeling. The corpus of this dataset comes from the Wall Street Journal. Moreover, this dataset contains 2,499 articles with a total of 1M words. In this experiments, we use three LSTM models, including 1-layer, 2-layer and 3-layer, to train this dataset.

In our experiments, we compare our proposed algorithm, FWAdaBound, with classical and latest proposed algorithms including OGD, Adam [14], and AdaBound [16]. Furthermore, the setting of parameters of all algorithms are shown as follows.

(i) OGD, the initial step size α is chosen from

$$\{1e-2, 5e-3, 1e-3, 5e-4, 1e-4\} \quad (65)$$

(ii) Adam [14], with $\beta_1 = 0.9, \beta_2 = 0.999, \alpha_t = \alpha/\sqrt{t}$, in which the initial step size α is chosen from

$$\{1e-2, 5e-3, 1e-3, 5e-4, 1e-4\} \quad (66)$$

Moreover, we use for the perturbation value $\varepsilon = 1e-8$.

(iii) AdaBound [16], with $\beta_1 = 0.9, \beta_2 = 0.999, \alpha_t = \alpha/\sqrt{t}$, in which the initial step size α is chosen from

$$\{1e-2, 5e-3, 1e-3, 5e-4, 1e-4\} \quad (67)$$

Moreover, $\varepsilon = 1e-8$.

(iv) FWAdaBound, our proposed algorithm, we directly applied the default hyperparameters of AdaBound (a learning rate of 0.001, $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\varepsilon = 1e-8$)

Next, we show the results of our experiments on CIFAR-10 and present the related analysis of the experimental results.

6.2. Experiment Results and Analysis

6.2.1. Image Classification. In the first experiment, we run the algorithms on CIFAR-10 for 200 epochs and measure the relationship between running time and average loss. It can be concluded from the first experiment that with the same epoch number, FWAdaBound spends the shortest time to complete the iterative task. It also confirms that FWAdaBound takes the least computation cost among all the experimental algorithms. The reason is that FWAdaBound uses linear search instead of the high-order projection steps in Adam and AdaBound. Moreover, Figure 1 shows the results for each algorithm on both ResNet-34 and DenseNet-121. In the part (a) of Figure 1, the average loss of our proposed algorithm FWAdaBound reaches the expected stable value in the shortest time in model ResNet-34. Similarly, in part (b) of Figure 1, FWAdaBound also takes the least time to reduce the average loss in model DenseNet-121. This suggests that with the same number of epochs, FWAdaBound spends the least time to complete the iteration task. Moreover, this validates that FWAdaBound takes the least computation cost among all the experimental algorithms. The reason is that FWAdaBound uses linear search instead of the high-order projection steps in Adam and AdaBound. Therefore, FWAdaBound can iterates much faster than Adam and AdaBound in each epoch.

Then, we execute the second experiment to verify the generalization ability of training accuracy. The results of the second experiment are shown in Figure 2. This figure shows that the training accuracy of FWAdaBound rises rapidly in the early stage of training and finally reaches the same height as AdaBound, which validates the generalization ability of training accuracy of FWAdaBound is better. Moreover, the figure also shows that the training accuracy of FWAdaBound is higher than that of Adam and AdaBound at each moment, which further indicates that the iteration cost of FWAdaBound is the lowest.

Finally, the last experiment verify the generalization ability of test accuracy of our proposed algorithm. And the results of this experiment are presented in Figure 3. Likewise, the test accuracy of FWAdaBound performs well in the early stage of iteration process and achieves the same performance as AdaBound in the final stage. Therefore, FWAdaBound also has a good generalization ability on test accuracy. In general, the generalization ability of FWAdaBound is the same as that of AdaBound but takes much less computation cost than AdaBound and Adam.

6.2.2. Language Modeling. In this group of experiments, we implement all the algorithms in 1-, 2-, and 3-layer LSTM models on the Penn Treebank dataset. The results of the experiments are shown in Figure 4 which presents the relationship between the perplexity and the running time of each algorithm. Note that the lower perplexity the better.

The experiment results of 1-layer LSTM show that FWAdaBound takes the least time to minimize the perplexity. On 1-layer LSTM model, FWAdaBound performs best in all algorithms, and Adam performs better than AdaBound.

In addition, in the experiments executed on 2-layer LSTM model, FWAdaBound has the quickest convergence rate among all the algorithms. Moreover, FWAdaBound takes 16.67% and 17.65% less running time than Adam and AdaBound, respectively. In this experiment, AdaBound performs better than Adam on the perplexity.

Finally, all the three algorithms are executed on the 3-layer LSTM model. The results show that FWAdaBound takes the least time to minimize the perplexity on Penn Treebank. In addition, FWAdaBound takes 11.83% and 14.05% less time than Adam and AdaBound, respectively. Moreover, AdaBound also performs better than Adam on the perplexity in this experiments.

7. Conclusion

In this paper, we proposed a Frank-Wolfe adaptive momentum online algorithm named FWAdaBound, which uses the Frank-Wolfe technique to avoid the projection operation. Moreover, our convergence analysis showed that the regret bound of FWAdaBound achieves to $O(T^{3/4})$, where T is a time horizon. In order to validate the performance of FWAdaBound in applications, we execute three groups of experiments for image classification and language modeling. The results show that FWAdaBound has good performance in the generalization ability of training and test accuracy.

Data Availability

The data that support the findings of this study are CIFAR-10 [21] and Penn Treebank datasets [22].

Conflicts of Interest

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China (NSFC) under Grant Nos. 62002102, 72002133, and 62176113, in part by the Ministry of Education of China Science Foundation under Grant No. 19YJC630174, in part by the Key Technologies R & D Program of Henan Province under Grant No. 212102210083, and in part by the Luoyang Major Scientic and Technological Innovation Projects under Grant No. 2101017A.

References

[1] J. Li, R. Feng, W. Sun, Z. Liu, and Q. Li, "QoE-driven coupled uplink and downlink rate adaptation for 360-degree video live streaming," *IEEE Communications Letters*, vol. 24, no. 4, pp. 863–867, 2020.

[2] J. Li, C. Zhang, Z. Liu, R. Hong, and H. Hu, "Optimal volumetric video streaming with hybrid saliency based tiling," *IEEE Transactions on Multimedia*, 2022.

[3] K. Ota, M. Dao, V. Mezaris, and F. De Natale, "Deep learning for mobile multimedia: a survey," *ACM Transactions on Multimedia Computing, Communications, and Applications*, vol. 13, no. 3s, article 34, pp. 1–22, 2017.

[4] Y. Zhou, M. Zhang, J. Zhu, R. Zheng, and Q. Wu, "A randomized block-coordinate Adam online learning optimization algorithm," *Neural Computing and Applications*, vol. 32, no. 16, pp. 12671–12684, 2020.

[5] Y. Zhou, X. Wang, M. Zhang, J. Zhu, R. Zheng, and Q. Wu, "MPCE: a maximum probability based cross entropy loss function for neural network classification," *IEEE Access*, vol. 7, pp. 146331–146341, 2019.

[6] Y. Lu, G. Lu, R. Lin, J. Li, and D. Zhang, "SRGC-Nets: sparse repeated group convolutional neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 8, pp. 2889–2902, 2020.

[7] P. Chriskos, C. Frantzidis, P. Gkivogkli, P. Bamidis, and C. Kourtidou-Papadeli, "Automatic sleep staging employing convolutional neural networks and cortical connectivity images," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 1, pp. 113–123, 2020.

[8] G. Wang, J. Qiao, J. Bi, Q. Jia, and M. Zhou, "An adaptive deep belief network with sparse restricted Boltzmann machines," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 10, pp. 4217–4228, 2020.

[9] J. Yu and X. Yan, "Whole process monitoring based on unstable neuron output information in hidden layers of deep belief network," *IEEE Transactions on Cybernetics*, vol. 50, no. 9, pp. 3998–4007, 2020.

[10] Y. Cao, R. Ji, L. Ji, G. Lei, H. Wang, and X. Shao, "l^2-MPTCP: a learning-driven latency-aware multipath transport scheme for industrial internet applications," *IEEE Transactions on Industrial Informatics*, 2022.

[11] J. 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.

[12] T. Tieleman and G. Hinton, "RMSprop: Divide the gradient by a running average if its recent magnitude," in *Neural Networks for Machine Learning*, pp. 23–31, Coursera, California, CA, USA, 2012.

[13] M. Zeiler, "ADADELTA: an adaptive learning rate method," 2012, CoRR abs/1212.5701, http://arxiv.org/abs/1212.5701.

[14] D. Kingma and J. Ba, "Adam: a method for stochastic optimization. ICLR (Poster)," 2015, http://arxiv.org/abs/1412.6980.

[15] S. Reddi, S. Kale, and S. Kumar, "On the convergence of Adam and beyond. International conference on learning representations," 2018, https://openreview.net/forum?id=ryQu7f-RZ.

[16] L. Luo, Y. Xiong, and Y. Liu, "Adaptive gradient methods with dynamic bound of learning rate. International conference on learning representations," 2019, https://openreview.net/forum?id=Bkg3g2R9FX.

[17] H. Huang, C. Wang, and B. Dong, "Nostalgic Adam: weighing more of the past gradients when designing the adaptive learning rate," 2018, CoRR abs/1805.07557, http://arxiv.org/abs/1805.07557.

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

[19] M. Jaggi, "Revisiting Frank-Wolfe: projection-free sparse convex optimization," *Proceedings of the 30th International Conference on Machine Learning*, vol. 28, no. 1, pp. 427–435, 2013.

[20] F. Huang, L. Tao, and S. Chen, "Accelerated stochastic gradient-free and projection-free methods," *Proceedings of the 37th International Conference on Machine Learning*, vol. 119, pp. 4519–4530, 2020.

[21] A. Krizhevsky, "Learning multiple layers of features from tiny images," 2009, Technical Report. Available: http://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf.

[22] M. Marcus, A. Bies, and B. Schasberger, "The Penn TreeBank: annotating predicate argument structure," in *Proc. the workshop on Human Language Technology*, Plainsboro, NJ, 1994.