OPT-AMSGrad: An Optimistic Acceleration of AMSGrad for Nonconvex Optimization

Anonymous Author(s)
Affiliation
Address
email

Abstract

In this paper, we propose a new variant of AMSGrad [30], a popular adaptive gradient based optimization algorithm widely used for training deep neural networks. Our algorithm adds prior knowledge about the sequence of consecutive mini-batch gradients and leverages its underlying structure making the gradients sequentially predictable. By exploiting the predictability and ideas from Optimistic Online Learning, the proposed algorithm can accelerate the convergence and increase sample efficiency. After establishing a tighter upper bound under some convexity conditions on the regret, we offer a complimentary view of our algorithm which generalizes the offline and stochastic version of nonconvex optimization. In the nonconvex case, we establish a non-asymptotic convergence bound independently of the initialization of the method. We illustrate the practical speedup on several deep learning models through numerical experiments.

1 Introduction

Deep learning models have been successful in several applications, from robotics (e.g. [21]), computer vision (e.g [18, 15]), reinforcement learning (e.g. [25]) and natural language processing (e.g. [16]). With the sheer size of modern datasets and the dimension of neural networks, speeding up training is of utmost importance. To do so, several algorithms have been proposed in recent years, such as AMSGRAD [30], ADAM [19], RMSPROP [34], ADADELTA [40], and NADAM [10].

All the prevalent algorithms for training deep networks mentioned above combine two ideas: the idea of adaptivity from ADAGRAD [11, 23] and the idea of momentum from NESTEROV'S METHOD [27] or HEAVY BALL method [28]. ADAGRAD is an online learning algorithm that works well compared to the standard online gradient descent when the gradient is sparse. Its update has a notable feature: it leverages an anisotropic learning rate depending on the magnitude of gradient in each dimension which helps in exploiting the geometry of the data. On the other hand, NESTEROV'S METHOD or HEAVY BALL Method [28] is an accelerated optimization algorithm which update not only depends on the current iterate and current gradient but also depends on the past gradients (i.e. momentum). State-of-the-art algorithms like AMSGRAD [30] and ADAM [19] leverage these ideas to accelerate the training of nonconvex objective functions such as deep neural networks losses.

In this paper, we propose an algorithm that goes further than the hybrid of the adaptivity and momentum approach. Our algorithm is inspired by OPTIMISTIC ONLINE LEARNING [7, 29, 33, 1, 24], which assumes that, in each round of online learning, a *predictable process* of the gradient of the loss function is available. Then an action is played exploiting these predictors. By capitalizing on this (possibly) arbitrary process, algorithms in OPTIMISTIC ONLINE LEARNING enjoy smaller regret than the ones without gradient predictions. We combine the OPTIMISTIC ONLINE LEARNING idea with the adaptivity and the momentum ideas to design a new algorithm — OPT-AMSGRAD.

A single work along that direction stands out. Daskalakis et al. [8] develop OPTIMISTIC-ADAM leveraging optimistic online mirror descent [29]. Yet, OPTIMISTIC-ADAM is specifically designed to optimize two-player games, e.g. GANs [15] which is in particular a two-player zero-sum game. There have been some related works in OPTIMISTIC ONLINE LEARNING [7, 29, 33] showing that if both players use an OPTIMISTIC type of update, then accelerating the convergence to the equilibrium of the game is possible. Daskalakis et al. [8] build on these related works and show that OPTIMISTIC-MIRROR-DESCENT can avoid the cycle behavior in a bilinear zero-sum game accelerating the convergence. In contrast, in this paper, the proposed algorithm is designed to accelerate nonconvex optimization (e.g. empirical risk minimization). To the best of our knowledge, this is the first work exploring towards this direction and bridging the unfilled *theoretical* gap at the crossroads of online learning and stochastic optimization. The contributions of this paper are as follows:

- We derive an optimistic variant of AMSGRAD borrowing techniques from online learning procedures. Our method relies on (I) the addition of *prior knowledge* in the sequence of the model parameter estimations alleviating a predictable process able to provide guesses of gradients through the iterations and (II) the construction of a *double update* algorithm done sequentially. We interpret this two-projection step as the learning of the global parameter and of an underlying scheme which makes the gradients sequentially predictable.
- We focus on the *theoretical* justifications of our method by establishing novel *non-asymptotic* and *global* convergence rates in both convex and nonconvex cases. Based on *convex regret minimization* and *nonconvex stochastic optimization* views, we prove, respectively, that our algorithm suffers regret of $\mathcal{O}(\sqrt{\sum_{t=1}^{T} \|g_t - m_t\|^2_{\psi_{t-1}}})$ and achieves a convergence rate $\mathcal{O}(\sqrt{d/T} + d/T)$, where g_t is the gradient and m_t is its prediction.

The proposed algorithm not only adapts to the informative dimensions, exhibits momentum, but also exploits a good guess of the next gradient to facilitate acceleration. Besides the global analysis of OPT-AMSGRAD, we conduct experiments and show that the proposed algorithm not only accelerates the training procedure, but also leads to better empirical generalization performance.

Section 2 is devoted to introductory notions on online learning for regret minimization and adaptive learning methods for nonconvex stochastic optimization. We introduce in Section 3 our new algorithm, namely OPT-AMSGRAD and provide a comprehensive global analysis in both *convex/online* and *nonconvex/offline* settings in Section 4. We illustrate the benefits of our method on several finite-sum nonconvex optimization problems in Section 5. The supplementary material of this paper is devoted to the proofs of our theoretical results.

Notations: We follow the notations of adaptive optimization [19, 30]. For any u, $v \in \mathbb{R}^d$, u/v represents the element-wise division, u^2 the element-wise square, $\sqrt{u}$ the element-wise square-root. We denote $g_{1:T}[i]$ as the sum of the i_{th} element of $g_1, \dots, g_T \in \mathbb{R}^d$ and $\| \cdot \|$ as the Euclidean norm.

2 Preliminaries

Optimistic Online learning. The standard setup of ONLINE LEARNING is that, in each round t, an online learner selects an action $w_t \in \Theta \subseteq \mathbb{R}^d$, observes $\ell_t(\cdot)$ and suffers the associated loss $\ell_t(w_t)$ after the action is committed. The goal of the learner is to minimize the regret,

$$\mathcal{R}_T(\{w_t\}) := \sum_{t=1}^{T} \ell_t(w_t) - \sum_{t=1}^{T} \ell_t(w^*) \ ,$$

which is the cumulative loss of the learner minus the cumulative loss of some benchmark $w^* \in \Theta$. The idea of OPTIMISTIC ONLINE LEARNING (e.g. [7, 29, 33, 1]) is as follows. In each round t, the learner exploits a guess $m_t(\cdot)$ of the gradient $\nabla \ell_t(\cdot)$ to choose an action w_t[1]. Consider the FOLLOW-THE-REGULARIZED-LEADER (FTRL, [17]) online learning algorithm which update reads

$$w_t = \arg\min_{w \in \Theta} \langle w, L_{t-1} \rangle + \tfrac{1}{\eta} \mathsf{R}(w) \ ,$$

where η is a parameter, $\mathsf{R}(\cdot)$ is a 1-strongly convex function with respect to a given norm on the constraint set Θ, and $L_{t-1} := \sum_{s=1}^{t-1} g_s$ is the cumulative sum of gradient vectors of the loss functions

[1] Imagine that if the learner would have known $\nabla \ell_t(\cdot)$ (*i.e.,* exact guess) before committing its action, then it would exploit the knowledge to determine its action and consequently minimize the regret.

up to round $t-1$. It has been shown that FTRL has regret at most $\mathcal{O}(\sqrt{\sum_{t=1}^T \|g_t\|_*^2})$. The update of its optimistic variant, noted OPTIMISTIC-FTRL and developed in [33] reads

$$w_t = \arg\min_{w\in\Theta} \langle w, L_{t-1} + m_t \rangle + \frac{1}{\eta} \mathsf{R}(w) \ , \tag{1}$$

where $\{m_t\}_{t>0}$ is a predictable process incorporating (possibly arbitrarily) knowledge about the sequence of gradients $\{g_t := \nabla \ell_t(w_t)\}_{t>0}$. Under the assumption that loss functions are convex, it has been shown in [33] that the regret of OPTIMISTIC-FTRL is at most $\mathcal{O}(\sqrt{\sum_{t=1}^T \|g_t - m_t\|_*^2})$.

Remark: Note that the usual worst-case bound is preserved even when the predictors $\{m_t\}_{t>0}$ do not predict well the gradients. Indeed, if we take the example of OPTIMISTIC-FTRL, the bound reads $\sqrt{\sum_{t=1}^T \|g_t - m_t\|_*^2} \leq 2 \max\limits_{w\in\Theta} \|\nabla \ell_t(w)\| \sqrt{T}$ which is equal to the usual bound up to a factor 2 [29] . Yet, when the predictors $\{m_t\}_{t>0}$ are well designed, the regret will be lower. We will have a similar argument when comparing OPT-AMSGRAD and AMSGRAD regret bounds in Section 4.1.

We emphasize, in Section 3, the importance of leveraging a good guess m_t for updating w_t in order to get a fast convergence rate (or equivalently, small regret) and introduce in Section 5 a simple predictable process $\{m_t\}_{t>0}$ leading to empirical acceleration on various applications.

Adaptive optimization methods. Adaptive optimization has been popular in various deep learning applications due to their superior empirical performance. ADAM [19], a popular adaptive algorithm, combines momentum [28] and anisotropic learning rate of ADAGRAD [11]. More specifically, the learning rate of ADAGRAD at time t for dimension j is proportional to the inverse of $\sqrt{\Sigma_{s=1}^t g_s[j]^2}$, where $g_s[j]$ is the j-th element of the gradient vector g_s at time s.

This adaptive learning rate helps accelerating the convergence when the gradient vector is sparse [11] but, when applying ADAGRAD to train deep neural networks, it is observed that the learning rate might decay too fast [19]. Therefore, Kingma and Ba [19] propose ADAM that uses a moving average of the gradients divided by the square root of the second moment of the moving average (element-wise multiplication), for updating the model parameter w. A variant, called AMSGRAD and detailed in Algorithm 1, has been developed in [30] to fix ADAM failures. The difference between ADAM and AMSGRAD lies in Line 7 of Algorithm 1. The AMSGRAD algorithm [30] applies the `max` operation on the second moment to guarantee a non-increasing learning rate $\eta_t/\sqrt{\hat{v}_t}$, which helps for the convergence (i.e. average regret $\mathcal{R}_T/T \to 0$).

Algorithm 1 AMSGRAD [30]

```
1: Required: parameter $\beta_1$, $\beta_2$, and $\eta_t$.
2: Init: $w_1 \in \Theta \subseteq \mathbb{R}^d$ and $v_0 = \epsilon 1 \in \mathbb{R}^d$.
3: for $t = 1$ to $T$ do
4:     Get mini-batch stochastic gradient $g_t$ at $w_t$.
5:     $\theta_t = \beta_1 \theta_{t-1} + (1-\beta_1) g_t$.
6:     $v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2$.
7:     $\hat{v}_t = \max(\hat{v}_{t-1}, v_t)$.
8:     $w_{t+1} = w_t - \eta_t \frac{\theta_t}{\sqrt{\hat{v}_t}}$. (element-wise division)
9: end for
```

3 OPT-AMSGRAD Algorithm

We formulate in this section the proposed optimistic acceleration of AMSGrad, namely OPT-AMSGRAD, and detailed in Algorithm 2. It combines the idea of adaptive optimization with optimistic learning. At each iteration, the learner computes a gradient vector $g_t := \nabla \ell_t(w_t)$ at w_t (line 4), then it maintains an exponential moving average of $\theta_t \in \mathbb{R}^d$ (line 5) and $v_t \in \mathbb{R}^d$ (line 6), which is followed by the `max` operation to get $\hat{v}_t \in \mathbb{R}^d$ (line 7). The learner first updates an auxiliary variable $\tilde{w}_{t+1} \in \Theta$ (line 8) and then computes the next model parameter w_{t+1} (line 9). Observe that the proposed algorithm does not reduce to AMSGRAD when $m_t = 0$, contrary to the optimistic variant of FTRL. Furthermore, combining line 8 and line 9 yields the following single update $w_{t+1} = \tilde{w}_t - \eta_t(\theta_t + h_{t+1})/\sqrt{\hat{v}_t}$.

Compared to AMSGRAD, the algorithm is characterized by a *two-level* update that interlinks some *auxiliary state* $\tilde{w}_t$ and the model parameter state, w_t, similarly to the OPTIMISTIC MIRROR DESCENT algorithm developed in [29]. It leverages the auxiliary variable (hidden model) to update and commit w_{t+1}, which exploits the guess m_{t+1}, see Figure 1. In the following analysis, we show that the interleaving actually leads to some cancellation in the regret bound. Such two-levels method where the guess m_t is equal to the last known gradient g_{t-1} has been exhibited recently in [7]. The gradient prediction process plays an important role as discussed in Section 5. The proposed OPT-AMSGRAD inherits three properties: (i) Adaptive learning rate of each dimension as ADAGRAD [11] (line 6, line 8 and line 9). (ii) Exponential moving average of the past gradients as NESTEROV'S METHOD [27] and the HEAVY-BALL method [28] (line 5). (iii) Optimistic update that exploits *prior knowledge* of the next gradient vector as in optimistic online learning algorithms [7, 29, 33] (line 9). The first property helps for acceleration when the gradient has a sparse structure. The second one is from the long-established idea of momentum which can also help for acceleration. The last one can lead to an acceleration when the prediction of the next gradient is good as mentioned above when introducing the regret bound for the OPTIMISTIC-FTRL algorithm. This property will be elaborated whilst establishing the theoretical analysis of OPT-AMSGRAD.

Algorithm 2 OPT-AMSGRAD

```
1: **Required**: parameter $\beta_1$, $\beta_2$, $\epsilon$, and $\eta_t$.
2: Init: $w_1 = w_{-1/2} \in \Theta \subseteq \mathbb{R}^d$ and $v_0 = \epsilon 1 \in \mathbb{R}^d$.
3: **for** $t = 1$ to $T$ **do**
4:     Get mini-batch stochastic gradient $g_t$ at $w_t$.
5:     $\theta_t = \beta_1 \theta_{t-1} + (1 - \beta_1) g_t$.
6:     $v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2$.
7:     $\hat{v}_t = \max(\hat{v}_{t-1}, v_t)$.
8:     $\tilde{w}_{t+1} = \tilde{w}_t - \eta_t \frac{\theta_t}{\sqrt{\hat{v}_t}}$.
9:     $w_{t+1} = \tilde{w}_{t+1} - \eta_t \frac{h_{t+1}}{\sqrt{\hat{v}_t}}$,
       where $h_{t+1} := \beta_1 \theta_{t-1} + (1 - \beta_1) m_{t+1}$ with
       $m_{t+1}$ the guess of $g_{t+1}$.
10: **end for**
```

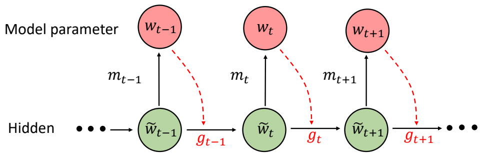

Figure 1: OPT-AMSGRAD Underlying Structure.

4 Global Convergence Analysis of OPT-AMSGRAD

More notations. We denote the Mahalanobis norm $\| \cdot \|_H := \sqrt{\langle \cdot, H \cdot \rangle}$ for some positive semidefinite (PSD) matrix H. We let $\psi_t(x) := \langle x, \text{diag}\{\hat{v}_t\}^{1/2} x \rangle$ for a PSD matrix $H_t^{1/2} := \text{diag}\{\hat{v}_t\}^{1/2}$, where $\text{diag}\{\hat{v}_t\}$ represents the diagonal matrix which i_{th} diagonal element is $\hat{v}_t[i]$ defined in Algorithm 2. We define its corresponding Mahalanobis norm $\| \cdot \|_{\psi_t} := \sqrt{\langle \cdot, \text{diag}\{\hat{v}_t\}^{1/2} \cdot \rangle}$, where we abuse the notation ψ_t to represent the PSD matrix $H_t^{1/2} := \text{diag}\{\hat{v}_t\}^{1/2}$. Note that $\psi_t(\cdot)$ is 1-strongly convex with respect to the norm $\| \cdot \|_{\psi_t}$. A consequence of 1-strongly convexity of $\psi_t(\cdot)$ is that $B_{\psi_t}(u, v) \geq \frac{1}{2} \|u - v\|_{\psi_t}^2$, where the Bregman divergence $B_{\psi_t}(u, v)$ is defined as $B_{\psi_t}(u, v) := \psi_t(u) - \psi_t(v) - \langle \psi_t(v), u - v \rangle$ with $\psi_t(\cdot)$ as the distance generating function. We also define the corresponding dual norm $\| \cdot \|_{\psi_t^*} := \sqrt{\langle \cdot, \text{diag}\{\hat{v}_t\}^{-1/2} \cdot \rangle}$.

4.1 Convex Regret Analysis

In this section, we assume convexity of $\{\ell_t\}_{t>0}$ and that Θ has a bounded diameter D_∞, which is a standard assumption for adaptive methods [30, 19] and is necessary in regret analysis.

Theorem 1. *Suppose the learner incurs a sequence of convex loss functions $\{\ell_t(\cdot)\}$. Then,* OPT-AMSGRAD *(Algorithm 2) has regret*

$$\mathcal{R}_T \leq \frac{B_{\psi_1}(w^*, \tilde{w}_1)}{\eta_1} + \sum_{t=1}^{T} \frac{\eta_t}{2} \|g_t - \tilde{m}_t\|_{\psi_{t-1}^*}^2 + \frac{D_\infty^2}{\eta_{\min}} \sum_{i=1}^{d} \hat{v}_T^{1/2}[i] + D_\infty^2 \beta_1^2 \sum_{t=1}^{T} \|g_t - \theta_{t-1}\|_{\psi_{t-1}^*} ,$$

where $\tilde{m}_{t+1} = \beta_1 \theta_{t-1} + (1 - \beta_1) m_{t+1}$, $g_t := \nabla \ell_t(w_t)$, $\eta_{\min} := \min_t \eta_t$ and D_∞^2 is the diameter of the bounded set Θ. The result holds for any benchmark $w^* \in \Theta$ and any step size sequence $\{\eta_t\}_{t>0}$.

Corollary 1. *Suppose $\beta_1 = 0$ and $\{v_t\}_{t>0}$ is a monotonically increasing sequence, then we obtain the following regret bound for any $w^* \in \Theta$ and sequence of stepsizes $\{\eta_t = \eta / \sqrt{t}\}_{t>0}$:*

$$\mathcal{R}_T \leq \frac{B_{\psi_1}}{\eta_1} + \frac{\eta \sqrt{1 + \log T}}{\sqrt{1 - \beta_2}} \sum_{i=1}^{d} \|(g - m)_{1:T}[i]\|_2 + \frac{D_\infty^2}{\eta_{\min}} \sum_{i=1}^{d} \left[(1 - \beta_2) \sum_{s=1}^{T} \beta_2^{T-s} g_s^2[i] \right]^{1/2} ,$$

where $B_{\psi_1} := B_{\psi_1}(w^*, \tilde{w}_1)$, $g_t := \nabla \ell_t(w_t)$ and $\eta_{\min} := \min_t \eta_t$.

We can compare the bound of Corollary 1 with that of AMSGRAD [30] with $\eta_t = \eta/\sqrt{t}$:

$$\mathcal{R}_T \leq \frac{\eta\sqrt{1+\log T}}{\sqrt{1-\beta_2}} \sum_{i=1}^{d} \|g_{1:T}[i]\|_2 + \frac{\sqrt{T}}{2\eta} D_\infty^2 \sum_{i=1}^{d} \hat{v}_T[i]^2 \ . \tag{2}$$

For convex regret minimization, Corollary 1 yields a regret of $\mathcal{O}(\sqrt{\sum_{t=1}^{T} \|g_t - m_t\|^2_{\psi^*_{t-1}}})$ with an access to an arbitrary predictable process $\{m_t\}_{t>0}$ of the mini-batch gradients. We notice from the second term in Corollary 1 compared to the first term in (2) that better predictors lead to lower regret. The construction of the predictions $\{m_t\}_{t>0}$ is thus of utmost importance for achieving optimal acceleration and can be learned through the iterations [29]. In Section 5, we derive a basic, yet effective, gradients prediction algorithm, see Algorithm 3, embedded in OPT-AMSGRAD.

4.2 Finite-Time Analysis in the Nonconvex Case

We discuss the offline and stochastic nonconvex optimization properties of our online framework. As stated in the Introduction, this paper is about solving optimization problems instead of solving zero-sum games. Classically, the optimization problem we are tackling reads:

$$\min_{w \in \Theta} f(w) := \mathbb{E}[f(w,\xi)] = n^{-1} \sum_{i=1}^{n} \mathbb{E}[f(w,\xi_i)] \ , \tag{3}$$

for a fixed batch of n samples $\{\xi_i\}_{i=1}^n$. The objective function $f(\cdot)$ is (potentially) nonconvex and has Lipschitz gradients. Set the terminating number, $T \in \{0, \dots, T_{\mathsf{M}} - 1\}$, as a discrete r.v. with:

$$P(T = \ell) = \frac{\eta_\ell}{\sum_{j=0}^{T_{\mathsf{M}}-1} \eta_j} \ , \tag{4}$$

where T_{M} is the maximum number of iteration. The random termination number (4) is inspired by [14] and is widely used for nonconvex optimization. Assume the following:

H1. *For any $t > 0$, the estimated parameter w_t stays within a ℓ_∞−ball. There exists a constant $W > 0$ such that $\|w_t\|_\infty \leq W$ almost surely.*

H2. *The function f is L-smooth (has L-Lipschitz gradients) w.r.t. the parameter w. There exists some constant $L > 0$ such that for $(w, \vartheta) \in \Theta^2$, $f(w) - f(\vartheta) - \nabla f(\vartheta)^\top (w - \vartheta) \leq \frac{L}{2} \|w - \vartheta\|^2$.*

We assume that the optimistic guess m_t at iteration t and the true gradient g_t are correlated:

H3. *There exists a constant $a \in \mathbb{R}$ such that for any $t > 0$, $0 < \langle m_t \,|\, g_t \rangle \leq a\|g_t\|^2$.*

We make a classical assumption in nonconvex optimization [14] on the magnitude of the gradient:

H4. *There exists a constant $\mathsf{M} > 0$ such that for any w and ξ, it holds $\|\nabla f(w,\xi)\| < \mathsf{M}$.*

We now derive important auxiliary Lemmas for our global analysis. The first one ensures bounded norms of quantities of interests (resulting from the bounded stochastic gradient assumption):

Lemma 1. *Assume H4, then the quantities defined in Algorithm 2 satisfy for any $w \in \Theta$ and $t > 0$, $\|\nabla f(w_t)\| < \mathsf{M}$, $\|\theta_t\| < \mathsf{M}$ and $\|\hat{v}_t\| < \mathsf{M}^2$.*

We now formulate the main result of our paper yielding a finite-time upper bound of the suboptimality condition $\mathbb{E}\left[\|\nabla f(w_T)\|^2\right]$ (set as the convergence criterion of interest, see [14]):

Theorem 2. *Assume H1-H4, $\beta_1 < \beta_2 \in [0,1)$ and a sequence of decreasing stepsizes $\{\eta_t\}_{t>0}$, then the following result holds:*

$$\mathbb{E}\left[\|\nabla f(w_T)\|_2^2\right] \leq \tilde{C}_1 \sqrt{\frac{d}{T_{\mathsf{M}}}} + \tilde{C}_2 \frac{1}{T_{\mathsf{M}}} \ ,$$

where T is a random termination number distributed according (4). The constants are defined as:

$$\tilde{C}_1 = \frac{\mathsf{M}}{(1-a\beta_1) + (\beta_1 + a)} \left[\frac{a(1-\beta_1)^2}{1-\beta_2} + 2L\frac{1}{1-\beta_2} + \Delta f + \frac{4L\beta_1^2(1+\beta_1^2)}{(1-\beta_1)(1-\beta_2)(1-\gamma)} \right]$$

$$\tilde{C}_2 = \frac{(a\beta_1^2 - 2a\beta_1 + \beta 1)\mathsf{M}^2}{(1-\beta_1)\left((1-a\beta_1) + (\beta_1 + a)\right)} \mathbb{E}\left[\left\|\hat{v}_0^{-1/2}\right\|\right] \quad where \quad \Delta f = f(\overline{w}_1) - f(\overline{w}_{T_{\mathsf{M}}+1}) \ .$$

The bound for our OPT-AMSGrad method matches the complexity bound of $\mathcal{O}(\sqrt{d/T_{\mathsf{M}}} + 1/T_{\mathsf{M}})$ of [14] for SGD considering the dependence of T only, and of [41] for AMSGrad method.

4.3 Checking H1 for a Deep Neural Network

As boundedness assumption H1 is generally hard to verify, we now show, for illustrative purposes, that the weights of a fully connected feed forward neural network stay in a bounded set when being trained using our method. The activation function for this section will be sigmoid function and we use a ℓ_2 regularization. We consider a fully connected feed forward neural network with L layers modeled by the function $\mathsf{MLN}(w, \xi) : \Theta^d \times \mathbb{R}^p \to \mathbb{R}$ defined as:

$$\mathsf{MLN}(w, \xi) = \sigma\left(w^{(L)}\sigma\left(w^{(L-1)} \dots \sigma\left(w^{(1)}\xi\right)\right)\right) \ , \tag{5}$$

where $w = [w^{(1)}, w^{(2)}, \cdots, w^{(L)}]$ is the vector of parameters, $\xi \in \mathbb{R}^p$ is the input data and σ is the sigmoid activation function. We assume a p dimension input data and a scalar output for simplicity. In this setting, the stochastic objective function (3) reads

$$f(w, \xi) = \mathcal{L}(\mathsf{MLN}(w, \xi), y) + \frac{\lambda}{2}\|w\|^2 \ ,$$

where $\mathcal{L}(\cdot, y)$ is the loss function (e.g., cross-entroy), y are the true labels and $\lambda > 0$ is the regularization parameter. We establish that assumption H1 is satisfied with a neural network as in (5):

Lemma 2. *Given the multilayer model* (5)*, assume the boundedness of the input data and of the loss function,* i.e., *for any $\xi \in \mathbb{R}^p$ and $y \in \mathbb{R}$ there is a constant $T > 0$ such that $\|\xi\| \leq 1$ a.s. and $|\mathcal{L}'(\cdot, y)| \leq T$ where $\mathcal{L}'(\cdot, y)$ denotes its derivative* w.r.t. *the parameter. Then for each layer $\ell \in [1, L]$, there exist a constant $A_{(\ell)}$ such that $\left\|w^{(\ell)}\right\| \leq A_{(\ell)}$*

5 Numerical Experiments

5.1 Gradient Estimation

From the analysis in the previous section, we understand that the choice of the prediction m_t plays an important role in the convergence of OPTIMISTIC-AMSGRAD. Some classical works in gradient prediction methods include ANDERSON acceleration [36], MINIMAL POLYNOMIAL EXTRAPOLATION [4], REDUCED RANK EXTRAPOLATION [12]. These methods aim at finding a fixed point g^* and assume that $\{g_t \in \mathbb{R}^d\}_{t>0}$ has the following linear relation:

$$g_t - g^* = A(g_{t-1} - g^*) + e_t, \tag{6}$$

where e_t is a second order term satisfying $\|e_t\|_2 = \mathcal{O}(\|g_{t-1} - g^*\|_2^2)$ and $A \in \mathbb{R}^{d \times d}$ is an unknown matrix, see [31] for details and results. For our numerical experiments, we run OPT-AMSGRAD using Algorithm 3 to construct the sequence $\{m_t\}_{t>0}$ which is based on estimating the limit of a sequence using the last iterates [3]. Specifically, at iteration t, m_t is obtained by (a) calling Algorithm 3 with a sequence of r past gradients, $\{g_{t-1}, g_{t-2}, \dots, g_{t-r}\}$ as input yielding the vector $c = [c_0, \dots, c_{r-1}]$ and (b) setting $m_t := \Sigma_{i=0}^{r-1} c_i g_{t-r+i}$. To see why the output from the extrapolation method may be a reasonable estimation, assume that the update converges to a stationary point (i.e. $g^* := \nabla f(w^*) = 0$ for the underlying function f). Then, we might rewrite (6) as $g_t = Ag_{t-1} + \mathcal{O}(\|g_{t-1}\|_2^2)u_{t-1}$, for some unit vector u_{t-1}. This equation suggests that the next gradient vector g_t is a linear transform of g_{t-1} plus an error vector that may not be in the span of A. If the algorithm converges to a stationary point, the magnitude of the error will converge to zero.

Algorithm 3 Regularized Approximated Minimal Polynomial Extrapolation [31]

```
Input: sequence $\{g_s \in \mathbb{R}^d\}_{s=0}^{s=r-1}$, parameter $\lambda > 0$.
Compute matrix $U = [g_1 - g_0, \dots, g_r - g_{r-1}] \in \mathbb{R}^{d \times r}$.
Obtain $z$ by solving $(U^\top U + \lambda I)z = \mathbf{1}$.
Get $c = z/(z^\top \mathbf{1})$.
Output: $\Sigma_{i=0}^{r-1} c_i g_i$, the approximation of the fixed point $g^*$.
```

Computational cost: This extrapolation step consists in: (a) Constructing the linear system $(U^\top U)$ which cost can be optimized to $\mathcal{O}(d)$, since the matrix U only changes one column at a time. (b) Solving the linear system which cost is $\mathcal{O}(r^3)$, and is negligible for a small r used in practice. (c) Outputting a weighted average of previous gradients which cost is $\mathcal{O}(r \times d)$ yielding a computational overhead of $\mathcal{O}\left((r+1)d + r^3\right)$. Yet, steps (a) and (c) are parallelizable in the final implementation.

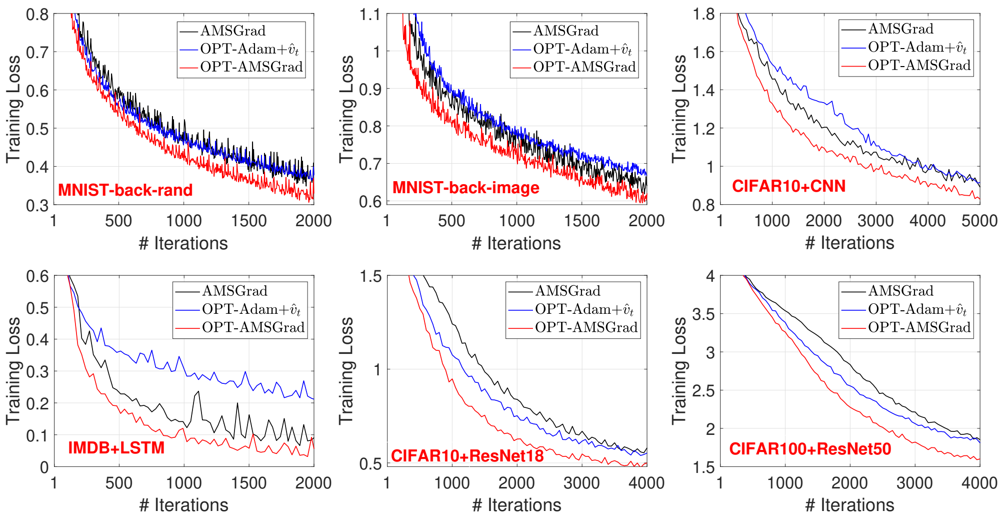

Figure 2: Training loss vs. Number of iterations for fully connected NN, CNN, LSTM and ResNet.

5.2 Classification Experiments

In this section, we provide experiments on classification tasks with various neural network architectures and datasets to demonstrate the effectiveness of OPT-AMSGRAD.

Methods. We consider two baselines. The first one is the original AMSGRAD. The hyper-parameters are set to be $\beta_1 = 0.9$ and $\beta_2 = 0.999$, see [30]. The other benchmark method is the OPTIMISTIC-ADAM+$\hat{v}_t$ [8], which details are given in the supplementary material. We use cross-entropy loss, a mini-batch size of 128 and tune the learning rates over a fine grid and report the best result for all methods. For OPT-AMSGRAD, we use $\beta_1 = 0.9$ and $\beta_2 = 0.999$ and the best step size η of AMSGRAD for a fair evaluation of the optimistic step. OPT-AMSGRAD has an additional parameter r that controls the number of previous gradients used for gradient prediction. We use $r = 5$ past gradient for empirical reasons, see Section 5.3. The algorithms are initialized at the same point and the results are averaged over 5 repetitions.

Datasets. Following [30] and [19], we compare different algorithms on *MNIST*, *CIFAR10*, *CIFAR100*, and *IMDB* datasets. For *MNIST*, we use two noisy variants namely *MNIST-back-rand* and *MNIST-back-image* from [20]. They both have 12 000 training samples and 50 000 test samples, where random background is inserted to the original *MNIST* hand-written digit images. For *MNIST-back-rand*, each image is inserted with a random background, which pixel values are generated uniformly from 0 to 255, while *MNIST-back-image* takes random patches from a black and white noisy background. The input dimension is 784 (28 × 28) and the number of classes is 10. *CIFAR10* and *CIFAR100* are popular computer-vision datasets of 50 000 training images and 10 000 test images, of size 32 × 32. The *IMDB* movie review dataset is a binary classification dataset with 25 000 training and testing samples respectively. It is a popular datasets for text classification.

Network architectures. We adopt a multi-layer fully connected neural network with hidden layers of 200 then 100 neurons (using ReLU activations and Softmax output) on *MNIST* variants. For CIFAR datasets, we adopt ALL-CNN network proposed by [32], built with convolutional blocks and dropout layers. In addition, we also apply residual networks, Resnet-18 and Resnet-50 [18], which have achieved state-of-the-art results. For the texture *IMDB* dataset, we consider a Long-Short Term Memory (LSTM) network [13] including a word embedding layer with 5 000 input entries representing most frequent words embedded into a 32 dimensional space. The output of the embedding layer is passed to 100 LSTM units then connected to 100 fully connected ReLU layers.

Results. Firstly, to illustrate the acceleration effect of OPT-AMSGRAD at early stage, we provide the training loss against number of iterations in Figure 2. We clearly observe that on all datasets, the proposed OPT-AMSGRAD converges faster than the other competing methods since fewer iterations are required to achieve the same precision, validating one of the main edges of OPT-AMSGRAD. We are also curious about the long-term performance and generalization of the proposed method in test phase. In Figure 3, we plot the results when the model is trained until the test accuracy stabilizes. We observe: (1) in the long term, OPT-AMSGRAD algorithm may converge to a better point with smaller objective function value, and (2) in these three applications, the proposed OPT-AMSGRAD also outperforms the competing methods in terms of test accuracy.

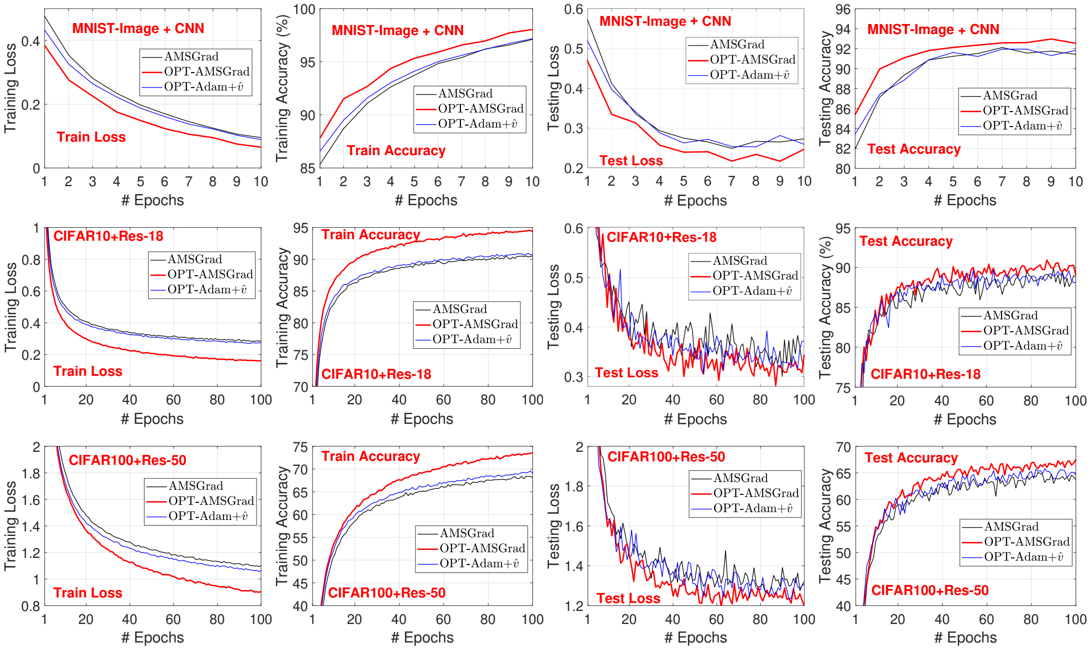

Figure 3: *MNIST-back-image* + CNN, *CIFAR10* + Res-18 and *CIFAR100* + Res-50 . We compare three methods in terms of training (cross-entropy) loss and accuracy, testing loss and accuracy.

5.3 Choice of parameter r

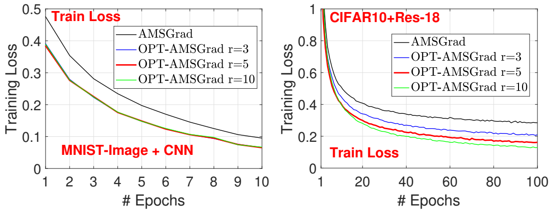

Figure 4: Training loss w.r.t. r.

Since the number of past gradients r is important in our algorithm, we compare Figure 4 the performance under different values $r = 3, 5, 10$ on two datasets. From the results we see that the choice of r does not have significant impact on the training loss. Taking into consideration both quality of gradient prediction and computational cost, $r = 5$ is a good choice for most applications. We remark that, empirically, the performance comparison among $r = 3, 5, 10$ is not absolutely consistent (i.e. more means better) in all cases. One possible reason is that for deep neural networks, the high diversity of computed gradients through the iterations, due to the highly nonconvex loss, makes them inefficient for sequentially building the predictable process $\{m_t\}_{t>0}$. Thus, only recent ones ($r \leq 5$) are used.

6 Conclusion

In this paper, we propose OPT-AMSGRAD, which combines optimistic online learning and AMSGRAD to improve sample efficiency and accelerate the process of training, in particular for deep neural networks. Given a good gradient prediction process, we demonstrate that the regret can be smaller than that of standard AMSGRAD. We also establish finite-time convergence bound on the second order moment of the gradient of the objective function matching that of state-of-the-art algorithms. Experiments on various deep learning problems demonstrate the effectiveness of the proposed algorithm in accelerating the empirical risk minimization procedure and empirically show better generalization properties of OPT-AMSGRAD.

7 Broader Impact

Broader Impact discussion is not applicable for this paper given the generality of both methods and numerical examples presented.

References

[1] J. Abernethy, K. A. Lai, K. Y. Levy, and J.-K. Wang. Faster rates for convex-concave games. *COLT*, 2018.

[2] N. Agarwal, B. Bullins, X. Chen, E. Hazan, K. Singh, C. Zhang, and Y. Zhang. Efficient full-matrix adaptive regularization. *ICML*, 2019.

[3] C. Brezinski and M. R. Zaglia. Extrapolation methods: theory and practice. *Elsevier*, 2013.

[4] S. Cabay and L. Jackson. A polynomial extrapolation method for finding limits and antilimits of vector sequences. *SIAM Journal on Numerical Analysis*, 1976.

[5] X. Chen, S. Liu, R. Sun, and M. Hong. On the convergence of a class of adam-type algorithms for non-convex optimization. *ICLR*, 2019.

[6] Z. Chen, Z. Yuan, J. Yi, B. Zhou, E. Chen, and T. Yang. Universal stagewise learning for non-convex problems with convergence on averaged solutions. *ICLR*, 2019.

[7] C.-K. Chiang, T. Yang, C.-J. Lee, M. Mahdavi, C.-J. Lu, R. Jin, and S. Zhu. Online optimization with gradual variations. *COLT*, 2012.

[8] C. Daskalakis, A. Ilyas, V. Syrgkanis, and H. Zeng. Training gans with optimism. *ICLR*, 2018.

[9] A. Défossez, L. Bottou, F. Bach, and N. Usunier. On the convergence of adam and adagrad. *arXiv preprint arXiv:2003.02395*, 2020.

[10] T. Dozat. Incorporating nesterov momentum into adam. *ICLR (Workshop Track)*, 2016.

[11] J. Duchi, E. Hazan, and Y. Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research (JMLR)*, 2011.

[12] R. Eddy. Extrapolating to the limit of a vector sequence. *Information linkage between applied mathematics and industry, Elsevier*, 1979.

[13] F. A. Gers, J. Schmidhuber, and F. Cummins. Learning to forget: Continual prediction with lstm. 1999.

[14] S. Ghadimi and G. Lan. Stochastic first-and zeroth-order methods for nonconvex stochastic programming. *SIAM Journal on Optimization*, 23(4):2341–2368, 2013.

[15] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. *NIPS*, 2014.

[16] A. Graves, A. rahman Mohamed, and G. Hinton. Speech recognition with deep recurrent neural networks. *ICASSP*, 2013.

[17] E. Hazan. Introduction to online convex optimization. *Foundations and Trends in Optimization*, 2016.

[18] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *CVPR*, 2016.

[19] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *ICLR*, 2015.

[20] H. Larochelle, D. Erhan, A. Courville, J. Bergstra, and Y. Bengio. An empirical evaluation of deep architectures on problems with many factors of variation. *ICML*, 2007.

[21] S. Levine, C. Finn, T. Darrell, and P. Abbeel. End-to-end training of deep visuomotor policies. *NIPS*, 2017.

[22] X. Li and F. Orabona. On the convergence of stochastic gradient descent with adaptive stepsizes. *AISTAT*, 2019.

[23] H. B. McMahan and M. J. Streeter. Adaptive bound optimization for online convex optimization. *COLT*, 2010.

[24] P. Mertikopoulos, B. Lecouat, H. Zenati, C.-S. Foo, V. Chandrasekhar, and G. Piliouras. Optimistic mirror descent in saddle-point problems: Going the extra (gradient) mile. *arXiv preprint arXiv:1807.02629*, 2018.

[25] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller. Playing atari with deep reinforcement learning. *NIPS (Deep Learning Workshop)*, 2013.

[26] M. Mohri and S. Yang. Accelerating optimization via adaptive prediction. *AISTATS*, 2016.

[27] Y. Nesterov. Introductory lectures on convex optimization: A basic course. *Springer*, 2004.

[28] B. T. Polyak. Some methods of speeding up the convergence of iteration methods. *Mathematics and Mathematical Physics*, 1964.

[29] S. Rakhlin and K. Sridharan. Optimization, learning, and games with predictable sequences. In *Advances in Neural Information Processing Systems*, pages 3066–3074, 2013.

[30] S. J. Reddi, S. Kale, and S. Kumar. On the convergence of adam and beyond. *ICLR*, 2018.

[31] D. Scieur, A. d'Aspremont, and F. Bach. Regularized nonlinear acceleration. *NIPS*, 2016.

[32] J. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller. Striving for simplicity: The all convolutional net. *ICLR*, 2015.

[33] V. Syrgkanis, A. Agarwal, H. Luo, and R. E. Schapire. Fast convergence of regularized learning in games. *NIPS*, 2015.

[34] T. Tieleman and G. Hinton. Rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural Networks for Machine Learning*, 2012.

[35] P. Tseng. On accelerated proximal gradient methods for convex-concave optimization. 2008.

[36] H. F. Walker and P. Ni. Anderson acceleration for fixed-point iterations. *SIAM Journal on Numerical Analysis*, 2011.

[37] R. Ward, X. Wu, and L. Bottou. Adagrad stepsizes: Sharp convergence over nonconvex landscapes, from any initialization. *ICML*, 2019.

[38] Y. Yan, T. Yang, Z. Li, Q. Lin, and Y. Yang. A unified analysis of stochastic momentum methods for deep learning. *arXiv preprint arXiv:1808.10396*, 2018.

[39] M. Zaheer, S. Reddi, D. Sachan, S. Kale, and S. Kumar. Adaptive methods for nonconvex optimization. *NeurIPS*, 2018.

[40] M. D. Zeiler. Adadelta: An adaptive learning rate method. *arXiv:1212.5701*, 2012.

[41] D. Zhou, Y. Tang, Z. Yang, Y. Cao, and Q. Gu. On the convergence of adaptive gradient methods for nonconvex optimization. *arXiv:1808.05671*, 2018.

[42] F. Zou and L. Shen. On the convergence of adagrad with momentum for training deep neural networks. *arXiv:1808.03408*, 2018.