ADAM$^+$: A STOCHASTIC METHOD WITH ADAPTIVE VARIANCE REDUCTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Adam is a widely used stochastic optimization method for deep learning applications. While practitioners prefer Adam because it requires less parameter tuning, its use is problematic from a theoretical point of view since it may not converge. Variants of Adam have been proposed with provable convergence guarantee, but they tend not be competitive with Adam on the practical performance. In this paper, we propose a new method named Adam$^+$ (pronounced as Adam-plus). Adam$^+$ retains some of the key components of Adam but it also has several noticeable differences: (i) it does not maintain the moving average of second moment estimate but instead computes the moving average of first moment estimate at extrapolated data points; (ii) its adaptive step size is formed not by dividing the square root of second moment estimate but instead by dividing the root of the norm of first moment estimate. As a result, Adam$^+$ requires few parameter tuning, as Adam, but it enjoys a provable convergence guarantee. Our analysis further shows that Adam$^+$ enjoys adaptive variance reduction, i.e., the variance of the stochastic gradient estimator reduces as the algorithm converges, hence enjoying an adaptive convergence. We also propose a more general variant of Adam$^+$ with different adaptive step sizes and establish their fast convergence rate. Our empirical studies on various deep learning tasks, including image classification, language modeling, and automatic speech recognition, demonstrate that Adam$^+$ significantly outperforms Adam and achieves comparable performance with best-tuned SGD and momentum SGD.

1 INTRODUCTION

Adaptive gradient methods (Duchi et al., 2011; McMahan & Streeter, 2010; Tieleman & Hinton, 2012; Kingma & Ba, 2014; Reddi et al., 2019) are one of the most important variants of Stochastic Gradient Descent (SGD) in modern machine learning applications. Contrary to SGD, adaptive gradient methods typically require little parameter tuning still retaining the computational efficiency of SGD. One of the most used adaptive methods is Adam (Kingma & Ba, 2014), which is considered by practitioners as the de-facto default optimizer for deep learning frameworks. Adam computes the update for every dimension of the model parameter through a moment estimation, i.e., the estimates of the first and second moments of the gradients. The estimates for first and second moments are updated using exponential moving averages with two different control parameters. These moving averages are the key difference between Adam and previous adaptive gradient methods, such as Adagrad (Duchi et al., 2011).

Although Adam exhibits great empirical performance, there still remain many mysteries about its convergence. First, it has been shown that Adam may not converge for some objective functions (Reddi et al., 2019; Chen et al., 2018b). Second, it is unclear what is the benefit that the moving average brings from theoretical point of view, especially its effect on the convergence rate. Third, it has been empirically observed that adaptive gradient methods can have worse generalization performance than its non-adaptive counterpart (e.g., SGD) on various deep learning tasks due to the coordinate-wise learning rates (Wilson et al., 2017).

The above issues motivate us to design a new algorithm which achieves the best of both worlds, i.e., provable convergence with benefits from the moving average and enjoying good generalization performance in deep learning. Specifically, we focus on the following optimization problem:

$$\min_{\mathbf{w}\in\mathbb{R}^d} F(\mathbf{w}),$$

Table 1: Summary of different algorithms with different assumptions and complexity results for finding an ϵ-stationary point. "Individual Smooth" means assuming that $F(\mathbf{w}) = \mathbb{E}_{\xi\sim\mathcal{D}}[f(\mathbf{w};\xi)]$ and that every component function $f(\mathbf{w};\xi)$ is L-smooth. "Hessian Lipschitz" means that $\|\nabla^2 F(\mathbf{x}) - \nabla^2 F(\mathbf{y})\| \leq L_H\|\mathbf{x}-\mathbf{y}\|$ holds for $\mathbf{x},\mathbf{y}$ and $L_H \geq 0$. "Type I" means that the complexity depends on $\mathbb{E}[\sum_{i=1}^T \|\mathbf{g}_{1:T,i}\|]$, where $\mathbf{g}_{1:T,i}$ stands for the i-th row of the matrix $[\mathbf{g}_1,\ldots,\mathbf{g}_T]$ with $\mathbf{g}_t$ being the stochastic gradient at t-th iteration and T being the number of iterations. "Type II" means that complexity depends on $\mathbb{E}[\sum_{t=1}^T \|\mathbf{z}_t\|]$, where $\mathbf{z}_t$ is the variance-reduced gradient estimator at t-th iteration.

Algorithm	Individual Smooth	Hessian Lipschitz	Worst-case Complexity better than $O(\epsilon^{-4})$?	Data-dependent Complexity
Generalized Adam (Chen et al., 2018b) PAdam (Chen et al., 2018a) Stagewise Adagrad (Chen et al., 2019) SGD with Adaptive Stepsize (Li & Orabona, 2019) Adagrad-Norm (Ward et al., 2019)	No	No	No	Type I
SPIDER (Fang et al., 2018) STORM (Cutkosky & Orabona, 2019) SNVRG (Zhou et al., 2018) Prox-SARAH (Pham et al., 2020)	Yes	No	Yes	N/A
SGD (Fang et al., 2019) Normalized momentum SGD (Cutkosky & Mehta, 2020)	No	Yes	Yes	N/A
Adam$^+$ (this work)	No	Yes	Yes	Type II

where we only have access to stochastic gradients of F. Note that F could possibly be nonconvex in $\mathbf{w}$. Due to the non-convexity, our goal is to design a stochastic first-order algorithm to find the ϵ-stationary point, i.e., finding $\mathbf{w}$ such that $\mathbb{E}\left[\|\nabla F(\mathbf{w})\|\right] \leq \epsilon$, with low iteration complexity.

Our key contribution is the design and analysis of a new stochastic method named Adam$^+$. Adam$^+$ retains some of the key components of Adam but it also has several noticeable differences: (i) it does not maintain the moving average of second moment estimate but instead computes the moving average of first moment estimate at extrapolated data points; (ii) its adaptive step size is formed not by dividing the square root of coordinate-wise second moment estimate but instead by dividing the root of the norm of first moment estimate. These features allow us to establish the adaptive convergence of Adam$^+$. Different from existing adaptive methods where the adaptive convergence depends on the growth rate of stochastic gradients (Duchi et al., 2011; McMahan & Streeter, 2010; Kingma & Ba, 2014; Luo et al., 2019; Reddi et al., 2019; Chen et al., 2018a;b; Ward et al., 2019; Li & Orabona, 2019; Chen et al., 2019), our adaptive convergence is due to the adaptive variance reduction property of our first order moment estimate. In existing literature, the variance reduction is usually achieved by large mini-batch (Goyal et al., 2017) or recursive variance reduction (Fang et al., 2018; Zhou et al., 2018; Pham et al., 2020; Cutkosky & Orabona, 2019). In contrast, we do not necessarily require large minibatch or computing stochastic gradients at two points per-iteration to achieve the variance reduction. In addition, we also establish a fast rate that matches the state-of-the-art complexity under the same conditions of a variant of Adam$^+$. Table 1 provides an overview of our results and a summary of existing results. We refer readers to Section F for a comprehensive survey of other related work. We further corroborate our theoretical results with an extensive empirical study on various deep learning tasks.

Our contributions are summarized below.

- We propose a new algorithm with adaptive step size, namely Adam$^+$, for general nonconvex optimization. We show that it enjoys a new type of data-dependent adaptive convergence that depends on the variance reduction property of first moment estimate. Notably, this data-dependent complexity does not require the presence of sparsity in stochastic gradients to guarantee fast convergence as in previous works (Duchi et al., 2011; Kingma & Ba, 2014; Reddi et al., 2019; Chen et al., 2019; 2018a). To the best of our knowledge, this is the first work establishing such new type of data-dependent complexity.
- We show that a general variant of our algorithm can achieve $O(\epsilon^{-3.5})$ worst-case complexity, which matches the state-of-the-art complexity guarantee under the Hessian Lipschitz assumption (Cutkosky & Mehta, 2020).
- We demonstrate the effectiveness of our algorithms on image classification, language modeling, and automatic speech recognition. Our empirical results show that our proposed algorithm consistently outperforms Adam on all tasks, and it achieves comparable performance with the best-tuned SGD and momentum SGD.

Algorithm 1 Adam$^+$: Good default settings for the tested machine learning problems are $\alpha = 0.1, a = 1, \beta = 0.1, \epsilon_0 = 10^{-8}$.

1: **Require:** $\alpha, a \geq 1$: stepsize parameters
2: **Require:** $\beta \in (0, 1)$: Exponential decay rates for the moment estimate
3: **Require:** $g_t(\mathbf{w})$: unbiased stochastic gradient with parameters $\mathbf{w}$ at iteration t
4: **Require:** $\mathbf{w}_0$: Initial parameter vector
5: $\mathbf{z}_0 = g_0(\mathbf{w}_0)$
6: **for** $t = 0, \ldots, T$ **do**
7: Set $\eta_t = \frac{\alpha\beta^a}{\max(\|\mathbf{z}_t\|^{1/2}, \epsilon_0)}$
8: $\mathbf{w}_{t+1} = \mathbf{w}_t - \eta_t \mathbf{z}_t$
9: $\widehat{\mathbf{w}}_{t+1} = (1 - 1/\beta)\mathbf{w}_t + 1/\beta \cdot \mathbf{w}_{t+1}$
10: $\mathbf{z}_{t+1} = (1 - \beta)\mathbf{z}_t + \beta g_{t+1}(\widehat{\mathbf{w}}_{t+1})$
11: **end for**

2 Algorithm and Theoretical Analysis

In this section, we introduce our algorithm Adam$^+$ (presented in Algorithm 1) and establish its convergence guarantees. Adam$^+$ resembles Adam in several aspects but also has noticeable differences. Similar to Adam, Adam$^+$ also maintains an exponential moving average of first moment (i.e., stochastic gradient), which is denoted by $\mathbf{z}_t$, and uses it for updating the solution in line 8. However, the difference is that the stochastic gradient is evaluated on an extrapolated data point $\widehat{\mathbf{w}}_{t+1}$, which is an extrapolation of two previous updates $\mathbf{w}_t$ and $\mathbf{w}_{t+1}$. Similar to Adam, Adam$^+$ also uses an adaptive step size that is proportional to $1/\|\mathbf{z}_t\|^{1/2}$. Nonetheless, the difference lies at its adaptive step size is directly computed from the square root of the norm of first moment estimate $\mathbf{z}_t$. In contrast, Adam uses an adaptive step size that is proportional to $1/\sqrt{\mathbf{v}_t}$, where $\mathbf{v}_t$ is an exponential moving average of second moment estimate. These two key components of Adam$^+$, i.e., extrapolation and adaptive step size from the root norm of the first moment estimate, make it enjoy two noticeable benefits: variance reduction of first moment estimate and adaptive convergence. We shall explain these two benefits later.

Before moving to the theoretical analysis, we would like to make some remarks. First, it is worth mentioning that the moving average estimate with extrapolation is inspired by the literature of stochastic compositional optimization (Wang et al., 2017). Wang et al. (2017) showed that the extrapolation helps balancing the noise in the gradients, reducing the bias in the estimates and giving a faster convergence rate. Here, our focus and analysis techniques are quite different. In fact, Wang et al. (2017) focuses on the compositional optimization while we consider a general nonconvex optimization setting. Moreover, the analysis in (Wang et al., 2017) mainly deals with the error of the gradient estimator caused by the compositional nature of the problem, while our analysis focuses on carefully designing adaptive normalization to obtain an adaptive and fast convergence rates. A similar extrapolation scheme has been also employed in the algorithm NIGT by Cutkosky & Mehta (2020). In later sections, we will also provide a more general variant of Adam$^+$ which subsumes NIGT as a special case.

Another important remark is that the update of Adam$^+$ is very different from the famous Nesterov's momentum method. In Nesterov's momentum method, the update of $\mathbf{w}_{t+1}$ uses the stochastic gradient at an extrapolated point $\widehat{\mathbf{w}}_{t+1} = \mathbf{w}_{t+1} + \gamma(\mathbf{w}_{t+1} - \mathbf{w}_t)$ with a momentum parameter $\gamma \in (0, 1)$. In contrast, in Adam$^+$ the update of $\mathbf{w}_{t+1}$ is using the moving average estimate at an extrapolated point $\widehat{\mathbf{w}}_{t+1} = \mathbf{w}_{t+1} + (1/\beta - 1)(\mathbf{w}_{t+1} - \mathbf{w}_t)$. Finally, Adam$^+$ does not employ coordinate-wise learning rates as in Adam, and hence it is expected to have better generalization performance according to Wilson et al. (2017).

2.1 Adaptive Variance Reduction and Adaptive Convergence

In this subsection, we analyze Adam$^+$ by showing its variance reduction property and adaptive convergence. To this end, we make the following assumptions.

Assumption 1. *There exists positive constants L, Δ, L_H, σ and an initial solution $\mathbf{w}_0$ such that*

(i) F is L-smooth, i.e., $\|\nabla F(\mathbf{x}) - \nabla F(\mathbf{y})\| \leq L \|\mathbf{x} - \mathbf{y}\|, \ \forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^d$.

(ii) For $\forall \mathbf{x} \in \mathbb{R}^d$, we have access to a first-order stochastic oracle at time t $g_t(\mathbf{x})$ such that $\mathbb{E}\left[g_t(\mathbf{x})\right] = \nabla F(\mathbf{x})$, $\mathbb{E}\left\|g_t(\mathbf{x}) - \nabla F(\mathbf{x})\right\|^2 \leq \sigma^2$.

(iii) ∇F is a L_H-smooth mapping, i.e., $\|\nabla^2 F(\mathbf{x}) - \nabla^2 F(\mathbf{y})\| \leq L_H \|\mathbf{x} - \mathbf{y}\|, \forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^d$.

(iv) $F(\mathbf{w}_0) - F_* \leq \Delta < \infty$, where $F_* = \inf_{\mathbf{w} \in \mathbb{R}^d} F(\mathbf{w})$.

Remark: Assumption 1 (i) and (ii), (iv) are standard assumptions made in literature of stochastic non-convex optimization (Ghadimi & Lan, 2013). Assumption (iii) is the assumption that deviates from typical analysis of stochastic methods. We leverage this assumption to explore the benefit of moving average, extrapolation and adaptive normalization. It is also used in some previous works for establishing fast rate of stochastic first-order methods for nonconvex optimization (Fang et al., 2019; Cutkosky & Mehta, 2020) and this assumption is essential to get fast rate due to the hardness result in (Arjevani et al., 2019). It is also the key assumption for finding a local minimum in previous works (Carmon et al., 2018; Agarwal et al., 2017; Jin et al., 2017).

We might also assume that the stochastic gradient estimator in Algorithm 1 satisfies the following variance property.

Assumption 2. *Assume that $\mathbb{E}[\|g_0(\mathbf{w}_0) - \nabla F(\mathbf{w}_0)\|^2] \leq \sigma_0^2$ and $\mathbb{E}[\|g_t(\mathbf{w}_t) - \nabla F(\mathbf{w}_t)\|^2] \leq \sigma_m^2, t \geq 1$.*

Remark: When g_0 (resp. g_t) is implemented by a mini-batch stochastic gradient with mini-batch size S, then σ_0^2 (resp. σ_m^2) can be set as σ^2/S by Assumption 1 (ii). We differentiate the initial variance and intermediate variance because they contribute differently to the convergence.

We first introduce a lemma to characterize the variance of the moving average gradient estimator $\mathbf{z}_t$.

Lemma 1. *Suppose Assumption 1 and Assumption 2 hold and $a \geq 1$. Then, there exists a sequence of random variables δ_t satisfying $\|\mathbf{z}_t - \nabla F(\mathbf{w}_t)\| \leq \delta_t$ for $\forall t \geq 0$,*

$$\begin{aligned}\mathbb{E}\left[\delta_{t+1}^2\right] &\leq \left(1 - \frac{\beta}{2}\right)\mathbb{E}\left[\delta_t^2\right] + 2\beta^2\sigma_m^2 + \mathbb{E}\left[\frac{CL_H^2\|\mathbf{w}_{t+1} - \mathbf{w}_t\|^4}{\beta^3}\right] \\ &\leq \left(1 - \frac{\beta}{2}\right)\mathbb{E}\left[\delta_t^2\right] + 2\beta^2\sigma_m^2 + \mathbb{E}\left[CL_H^2\alpha^4\beta^{4a-3}\|\mathbf{z}_t\|^2\right],\end{aligned}$$

where $C = 1944$.

Remark: Note that δ_t is an upper bound of $\|\mathbf{z}_t - \nabla F(\mathbf{w}_t)\|$, the above lemma can be used to illustrate the variance reduction effect for the gradient estimator $\mathbf{z}_t$. To this end, we can bound $\|\mathbf{z}_t\|^2 \leq 2\delta_t^2 + 2\|\nabla F(\mathbf{w}_t)\|^2$, then the term $CL^2\alpha^4\beta^{4a-3}\delta_t^2$ can be canceled with $-\beta/4\delta_t^2$ with small enough α. Hence, we have $\mathbb{E}\delta_{t+1}^2 \leq (1 - \beta/4)\mathbb{E}[\delta_t^2] + 2\beta^2\sigma_m^2 + c\mathbb{E}[\|\nabla F(\mathbf{w}_t)\|^2]$ with a small constant c. As the algorithm converges with $\mathbb{E}[\|\nabla F(\mathbf{w}_t)\|^2]$ and β decreases to zero, the variance of $\mathbf{z}_t$ will also decrease. Indeed, the above recursion of $\mathbf{z}_t$'s variance resembles that of the recursive variance reduced gradient estimators (e.g., SPIDER (Fang et al., 2018), STORM (Cutkosky & Orabona, 2019)). The benefit of using Adam$^+$ is that we do not need to compute stochastic gradient twice at each iteration.

We can now state our convergence rates for Algorithm 1.

Theorem 1. *Suppose Assumption 1 and Assumption 2 hold. Suppose $\|\nabla F(\mathbf{w})\| \leq G$ for any $\mathbf{w} \in \mathbb{R}^d$. By choosing the parameters such that $\alpha^4 \leq \frac{1}{36CL_H^2}$, $\alpha \leq \frac{1}{4L}$, $a = 1$ and $\epsilon_0 = \beta^a$, we have*

$$\frac{1}{T}\sum_{t=1}^{T}\mathbb{E}\left\|\nabla F(\mathbf{w}_t)\right\|^2 \leq \frac{G\mathbb{E}\left[\sum_{t=1}^{T}\|\mathbf{z}_t\|\right]}{T} + \frac{\Delta}{\alpha T} + \frac{18\sigma_0^2}{\beta T} + 30\beta\sigma_m^2 . \tag{1}$$

In addition, suppose the initial batch size is T_0 and the intermediate batch size is m, and choose $\beta = T^{-b}$ with $0 \leq b \leq 1$, we have

$$\frac{1}{T}\sum_{t=1}^{T}\mathbb{E}\left\|\nabla F(\mathbf{w}_t)\right\|^2 \leq \frac{\mathbb{E}\left[G\sum_{t=1}^{T}\|\mathbf{z}_t\|\right]}{T} + \frac{\Delta}{\alpha T} + \frac{18\sigma^2}{T^{1-b}T_0} + \frac{30\sigma^2}{mT^b} . \tag{2}$$

Theorem 2. *Suppose Assumption 1 and Assumption 2 hold. By choosing parameters such that* $640\alpha^3 L_H^{3/2} \leq 1/120$, $a = 1, \epsilon_0 = 0, \beta = 1/T^s$ *with* $0 < s < 1$, *then it takes* $T = O\left(poly(1/\epsilon)\right)$ *number of iterations to ensure that*

$$\frac{1}{T}\sum_{t=1}^{T}\mathbb{E}\left[\|\nabla F(\mathbf{w}_t)\|^{3/2}\right] \leq \epsilon^{3/2}, \quad \frac{1}{T}\mathbb{E}\left[\sum_{t=1}^{T}\delta_t^{3/2}\right] \leq \epsilon^{3/2}.$$

Remarks:

- From Theorem 1, we can observe that the convergence rate of Adam$^+$ crucially depends on the growth rate of $\mathbb{E}\left[\sum_{t=1}^{T}\|\mathbf{z}_t\|\right]$, which gives a data-dependent adaptive complexity. If $\mathbb{E}\left[\sum_{t=1}^{T}\|\mathbf{z}_t\|\right] \leq T^{\alpha}$ with $\alpha < 1$, then the algorithm converges. Smaller α implies faster convergence. Our goal is to ensure that $\frac{1}{T}\sum_{t=1}^{T}\mathbb{E}\|\nabla F(\mathbf{w}_t)\|^2 \leq \epsilon^2$. Choosing $b = 1-\alpha$, $m = O(1)$ and $T_0 = T^{1-\alpha} = O(\epsilon^{-2})$, and we end up with $T = O\left(\epsilon^{-\frac{2}{1-\alpha}}\right)$ complexity.

- Theorem 2 shows that in the ergodic sense, the Algorithm Adam$^+$ always converges, and the variance gets smaller when the number of iteration gets larger. Theorem 2 rules out the case that the magnitude of $\mathbf{z}_t$ converges to a constant and the bound (2) in Theorem 1 becomes vacuous.

- To compare with Adam-style algorithms (e.g., Adam, AdaGrad), these algorithms' convergence depend on the growth rate of stochastic gradient, i.e., $\sum_{i=1}^{d}\|\mathbf{g}_{1:T,i}\|/T$, where $\mathbf{g}_{1:T,i} = [g_{1,i}, \ldots, g_{T,i}]$ denotes the i-th coordinate of all historical stochastic gradients. Hence, the data determines the growth rate of stochastic gradient. If the stochastic gradients are not sparse, then its growth rate may not be slow and these Adam-style algorithms may suffer from slow convergence. In contrast, for Adam$^+$ the convergence can be accelerated by the variance reduction property. Note that we have $\mathbb{E}\left[\sum_{t=1}^{T}\|\mathbf{z}_t\|\right]/T \leq \mathbb{E}\left[\sum_{t=1}^{T}(\delta_t + \|\nabla F(\mathbf{w}_t)\|)\right]/T$. Hence, Adam$^+$'s convergence depends on the variance reduction property of $\mathbf{z}_t$.

2.2 A General Variant of Adam$^+$: Fast Convergence with Large Mini-batch

Next, we introduce a more general variant of Adam$^+$ by making a simple change. In particular, we keep all steps the same as in Algorithm 1 except the adaptive step size is now set as $\eta_t = \frac{\alpha\beta^a}{\max(\|\mathbf{z}_t\|^p, \epsilon_0)}$, where $p \in [1/2, 1)$ is parameter. We refer to this general variant of Adam$^+$ as power normalized Adam$^+$ (Nadam$^+$). This generalization allows us to compare with some existing methods and to establish fast convergence rate. First, we notice that when setting $p = 1$ and $a = 5/4$ and $\beta = 1/T^{4/7}$, Nadam$^+$ is almost the same as the stochastic method NIGT (Cutkosky & Mehta, 2020) with only some minor differences. However, we observed that normalizing by $\|\mathbf{z}_t\|$ leads to slow convergence in practice, so we are instead interested in $p < 1$. Below, we will show that NAdam$^+$ with $p < 1$ can achieve a fast rate of $1/\epsilon^{3.5}$, which is the same as NIGT.

Theorem 3. *Under the same assumption as in Theorem 1, further assume* $\sigma_0^2 = \sigma^2/T_0$ *and* $\sigma_m^2 = \sigma^2/m$. *By using the step size* $\eta_t = \frac{\alpha\beta^{4/3}}{\max\left(\|\mathbf{z}_t\|^{2/3}, \epsilon_0\right)}$ *in Algorithm 1 with* $CL^2\alpha^4 \leq 1/14$, $\epsilon_0 = 2\beta^{4/3}$, *in order to have* $\mathbb{E}\left[\|\nabla F(\mathbf{w}_\tau)\|\right] \leq \epsilon$ *for a randomly selected solution* $\mathbf{w}_\tau$ *from* $\{\mathbf{w}_1, \ldots, \mathbf{w}_T\}$, *it suffice to set* $\beta = O(\epsilon^{1/2})$, $T = O(\epsilon^{-2})$, *the initial batch size* $T_0 = 1/\beta = O(\epsilon^{-1/2})$, *the intermediate batch size as* $m = 1/\beta^3 = O(\epsilon^{-3/2})$, *which ends up with the total complexity* $O(\epsilon^{-3.5})$.

Remark: Note that the above theorem establishes the fast convergence rate for Nadam$^+$ with $p = 2/3$. Indeed, we can also establish a fast rate of Adam$^+$ (where $p = 1/2$) in the order of $O(1/\epsilon^{3.625})$ with details provided in the Appendix E.

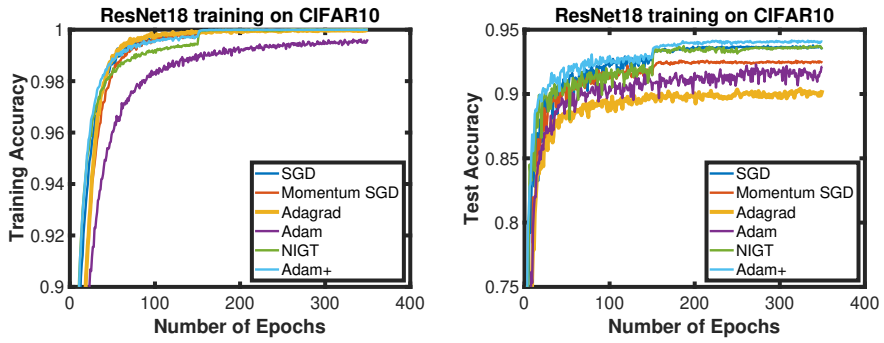

Figure 1: Comparison of optimization methods for ResNet18 Training on CIFAR10.

3 EXPERIMENTS

In this section, we conduct empirical studies to verify the effectiveness of the proposed algorithm on three different tasks: image classification on CIFAR10 and CIFAR100 dataset (Krizhevsky et al., 2009), language modeling on Wiki-Text2 dataset (Merity, 2016) and automatic speech recognition on SWB-300 dataset (Saon et al., 2017). We choose tasks from different domains to demonstrate the applicability for the real-world deep learning tasks in a broad sense. The detailed description is presented in Table 2. We compare our algorithm Adam$^+$ with SGD, momentum SGD, Adagrad, NIGT and Adam. We choose the same random initialization for each algorithm, and run a fixed number of epochs for every task.

Table 2: Summary of setups in the experiments.

Domain	Task	Architecture	Dataset
Computer Vision	Image Classification	ResNet18	CIFAR10
Computer Vision	Image Classification	VGG19	CIFAR100
Natural Language Processing	Language Modeling	Two-layer LSTM	Wiki-Text2
Automatic Speech Recognition	Speech Recognition	Six-layer BiLSTM	SWB-300

3.1 IMAGE CLASSIFICATION

CIFAR10 and CIFAR100 In the first experiment, we consider training ResNet18 (He et al., 2016) and VGG19 (Simonyan & Zisserman, 2014) to do image classification task on CIFAR10 and CIFAR100 dataset respectively. For every optimizer, we use batch size 128 and run 350 epochs. For SGD and momentum SGD, we set the initial learning rate to be 0.1 for the first 150 epochs, and the learning rate is decreased by a factor of 10 for every 100 epochs. For Adagrad and Adam, the initial learning rate is tuned from $\{0.1, 0.01, 0.001\}$ and we choose the one with the best performance. For NIGT, we tune the their momentum parameter from $\{0.01, 0.1, 0.9\}$ and the learning rate is chosen the same as in SGD. For Adam$^+$, the learning rate is set according to Algorithm 1, in which we choose $\beta = 0.1$ and the value of α is the same as the learning rate used in SGD. We report training and test accuracy versus the number of epochs in Figure 1 for CIFAR10 and Figure 2 for CIFAR100. We observe that our algorithm consistently outperforms all other algorithms on both CIFAR10 and CIFAR100, in terms of both training and testing accuracy. Notably, we have some interesting observations for the training of VGG19 on CIFAR100. First, both Adam$^+$ and NIGT significantly outperform SGD, momentum SGD, Adagrad and Adam. Second, Adam$^+$ achieves almost the same final accuracy as NIGT, and Adam$^+$ converges much faster in the early stage of the training.

3.2 LANGUAGE MODELING

Wiki-text2 In the second experiment, we consider the language modeling task on WikiText-2 dataset. We use a 2-layer LSTM (Hochreiter & Schmidhuber, 1997). The size of word embeddings is 650 and the number of hidden units per layer is 650. We run every algorithm for 40 epochs, with batch size 20 and dropout ratio 0.5. For SGD and momentum SGD, we tune the initial learning rate

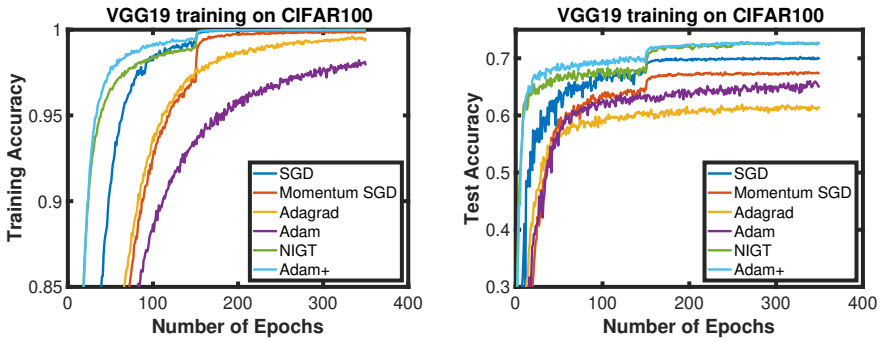

Figure 2: Comparison of optimization methods for VGG19 training on CIFAR100.

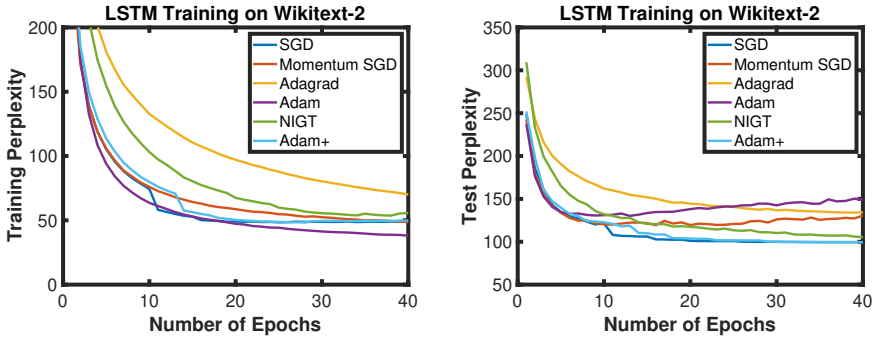

Figure 3: Comparison of optimization methods for two-layers LSTM training on WikiText-2.

from $\{0.1, 0.2, 0.5, 5, 10, 20\}$ and decrease the learning rate by factor of 4 when the validation error saturates. For Adagrad and Adam, we tune the initial learning rate from $\{0.001, 0.01, 0.1, 1.0\}$. We report the best performance for these methods across the range of learning rate. For NIGT, we tune the initial value of learning rate from the same range as in SGD, and tune the momentum parameter β from $\{0.01, 0.1, 0.9\}$. The learning rate and β are both decreased by a factor of 4 when the validation error saturates. For Adam$^+$, we follow the same tuning strategy as NIGT.

We report both training and test perplexity versus the number of epochs in Figure 3. From the Figure, we have the following observations: First, in terms of training perplexity, our algorithm achieves comparable performance with SGD and momentum SGD and outperforms Adagrad and NIGT, and it is worse than Adam. Second, in terms of test perplexity, our algorithm outperforms Adam, Adagrad, NIGT and momentum SGD, and it is comparable to SGD. An interesting observation is that Adam does not generalize well even if it has fast convergence in terms of training error, which is consistent with the observations in (Wilson et al., 2017).

3.3 Automatic Speech Recognition

SWB-300 In the third experiment, we consider the automatic speech recognition task on SWB-300 dataset (Saon et al., 2017). SWB-300 contains roughly 300 hours of training data of over 4 million samples (30GB) and roughly 6 hours of held-out data of over 0.08 million samples (0.6GB). Each training sample is a fusion of FMLLR (40-dim), i-Vector (100-dim), and logmel with its delta and double delta. The acoustic model is a long short-term memory (LSTM) model with 6 bi-directional layers. Each layer contains 1,024 cells (512 cells in each direction). On top of the LSTM layers, there is a linear projection layer with 256 hidden units, followed by a softmax output layer with 32,000 (i.e., 32,000 classes) units corresponding to context-dependent HMM states. The LSTM is unrolled with 21 frames and trained with non-overlapping feature sub-sequences of that length. This model contains over 43 million parameters and is about 165MB large. The training takes about 20 hours on 1 V100 GPU. To compare, we adopt the well-tuned Momentum SGD strategy as described in (Zhang et al., 2019) for this task as the baseline: batch size is 256, learning rate is

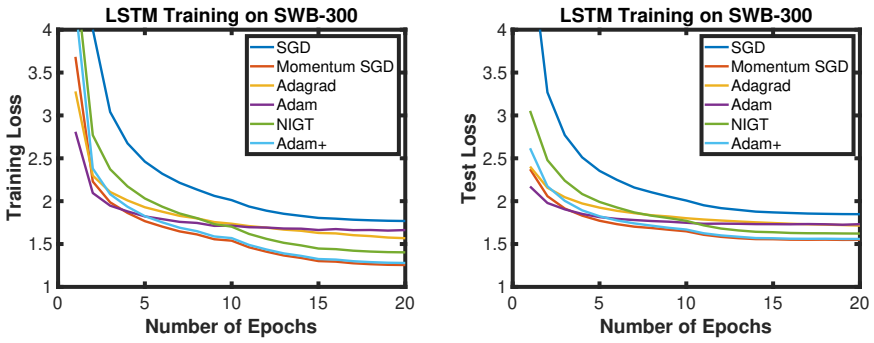

Figure 4: Comparison of optimization methods for six-layers LSTM training on SWB-300.

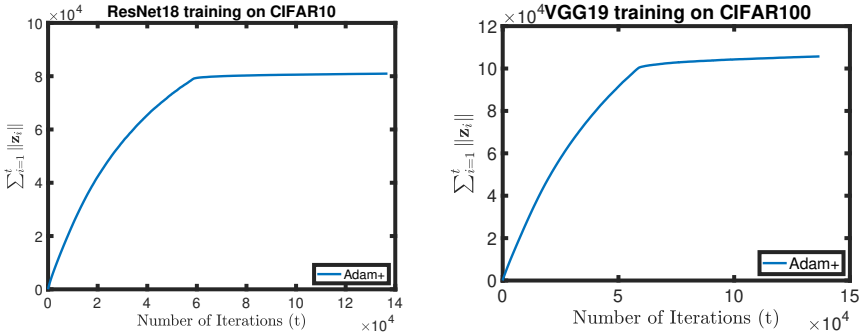

Figure 5: The growth of quantity $\sum_{i=1}^{t} \|\mathbf{z}_i\|$ in Adam$^+$

0.1 for the first 10 epochs and then annealed by $\sqrt{0.5}$ for another 10 epochs, with momentum 0.9. We grid search the learning rate of Adam and Adagrad from $\{0.1, 0.01, 0.001\}$, and report the best configuration we have found (Adam with learning rate 0.001 and Adagrad with learning rate 0.01). For NIGT, we also follow the same learning rate setup (including annealing) as in Momentum SGD baseline. In addition, we fine tuned β in NIGT by exploring β in $\{0.01, 0.1, 0.9\}$ and reported the best configuration (β=0.9). For Adam$^+$, we follow the same learning rate and annealing strategy as in the Momentum SGD and tuned β in the same way as in NGIT, reporting the best configuration (β=0.01). From Figure 4, Adam$^+$ achieves the indistinguishable training loss and held-out loss w.r.t. well-tuned Momentum SGD baseline and significantly outperforms the other optimizers.

3.4 Growth Rate of $\sum_{i=1}^{t} \|\mathbf{z}_i\|$

In this subsection, we consider the growth rate of $\sum_{i=1}^{t} \|\mathbf{z}_i\|$, since they crucially affect the convergence rate as shown in Theorem 1. We report the results of both ResNet18 training on CIFAR10 dataset and VGG19 training on CIFAR100 dataset. From Figure 5, we can observe that it quickly reaches a plateau and then grows at a very slow rate with respect to the number of iterations. This phenomenon verifies the variance reduction effect and also explains the reason why Adam$^+$ enjoys a fast convergence speed in practice.

4 Conclusion

In this paper, we design a new algorithm named Adam$^+$ to train deep neural networks efficiently. Different from Adam, Adam$^+$ updates the solution using moving average of stochastic gradients calculated at the extrapolated points and adaptive normalization on only first-order statistics of stochastic gradients. We establish data-dependent adaptive complexity results for Adam$^+$ from the perspective of adaptive variance reduction, and also show that a variant of Adam$^+$ achieves state-of-the-art complexity. Extensive empirical studies on several tasks verify the effectiveness of the proposed algorithm. We also empirically show that the slow growth rate of the new gradient estimator, providing the reason why Adam$^+$ enjoys fast convergence in practice.

REFERENCES

Naman Agarwal, Zeyuan Allen-Zhu, Brian Bullins, Elad Hazan, and Tengyu Ma. Finding approximate local minima faster than gradient descent. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pp. 1195–1199, 2017.

Zeyuan Allen-Zhu and Elad Hazan. Variance reduction for faster non-convex optimization. In *International conference on machine learning*, pp. 699–707, 2016.

Yossi Arjevani, Yair Carmon, John C Duchi, Dylan J Foster, Nathan Srebro, and Blake Woodworth. Lower bounds for non-convex stochastic optimization. *arXiv preprint arXiv:1912.02365*, 2019.

Yair Carmon, John C Duchi, Oliver Hinder, and Aaron Sidford. Accelerated methods for nonconvex optimization. *SIAM Journal on Optimization*, 28(2):1751–1772, 2018.

Jinghui Chen, Dongruo Zhou, Yiqi Tang, Ziyan Yang, and Quanquan Gu. Closing the generalization gap of adaptive gradient methods in training deep neural networks. *arXiv preprint arXiv:1806.06763*, 2018a.

Xiangyi Chen, Sijia Liu, Ruoyu Sun, and Mingyi Hong. On the convergence of a class of Adam-type algorithms for non-convex optimization. *arXiv preprint arXiv:1808.02941*, 2018b.

Zaiyi Chen, Zhuoning Yuan, Jinfeng Yi, Bowen Zhou, Enhong Chen, and Tianbao Yang. Universal stagewise learning for non-convex problems with convergence on averaged solutions. In *International Conference on Learning Representations*, 2019.

Ashok Cutkosky and Harsh Mehta. Momentum improves normalized SGD. *arXiv preprint arXiv:2002.03305*, 2020.

Ashok Cutkosky and Francesco Orabona. Momentum-based variance reduction in non-convex SGD. In *Advances in Neural Information Processing Systems*, pp. 15236–15245, 2019.

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul):2121–2159, 2011.

Cong Fang, Chris Junchi Li, Zhouchen Lin, and Tong Zhang. Spider: Near-optimal non-convex optimization via stochastic path-integrated differential estimator. In *Advances in Neural Information Processing Systems*, pp. 689–699, 2018.

Cong Fang, Zhouchen Lin, and Tong Zhang. Sharp analysis for nonconvex SGD escaping from saddle points. *arXiv preprint arXiv:1902.00247*, 2019.

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

Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. Accurate, large minibatch SGD: Training imagenet in 1 hour. *arXiv preprint arXiv:1706.02677*, 2017.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8): 1735–1780, 1997.

Chi Jin, Rong Ge, Praneeth Netrapalli, Sham M Kakade, and Michael I Jordan. How to escape saddle points efficiently. *arXiv preprint arXiv:1703.00887*, 2017.

Rie Johnson and Tong Zhang. Accelerating stochastic gradient descent using predictive variance reduction. In *Advances in neural information processing systems*, pp. 315–323, 2013.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. CIFAR-10 and CIFAR-100 datasets. *URl: https://www. cs. toronto. edu/kriz/cifar. html*, 6:1, 2009.

Lihua Lei, Cheng Ju, Jianbo Chen, and Michael I Jordan. Non-convex finite-sum optimization via SCSG methods. In *Advances in Neural Information Processing Systems*, pp. 2348–2358, 2017.

Xiaoyu Li and Francesco Orabona. On the convergence of stochastic gradient descent with adaptive stepsizes. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pp. 983–992, 2019.

Liangchen Luo, Yuanhao Xiong, Yan Liu, and Xu Sun. Adaptive gradient methods with dynamic bound of learning rate. *arXiv preprint arXiv:1902.09843*, 2019.

H Brendan McMahan and Matthew Streeter. Adaptive bound optimization for online convex optimization. *arXiv preprint arXiv:1002.4908*, 2010.

Stephen Merity. The WikiText long term dependency language modeling dataset. *Salesforce Metamind*, 9, 2016.

Nhan H Pham, Lam M Nguyen, Dzung T Phan, and Quoc Tran-Dinh. ProxSARAH: An efficient algorithmic framework for stochastic composite nonconvex optimization. *Journal of Machine Learning Research*, 21(110):1–48, 2020.

Sashank J Reddi, Ahmed Hefny, Suvrit Sra, Barnabas Poczos, and Alex Smola. Stochastic variance reduction for nonconvex optimization. In *International conference on machine learning*, pp. 314–323, 2016.

Sashank J Reddi, Satyen Kale, and Sanjiv Kumar. On the convergence of Adam and beyond. *arXiv preprint arXiv:1904.09237*, 2019.

George Saon, Gakuto Kurata, Tom Sercu, Kartik Audhkhasi, Samuel Thomas, Dimitrios Dimitriadis, Xiaodong Cui, Bhuvana Ramabhadran, Michael Picheny, Lynn-Li Lim, Bergul Roomi, and Phil Hall. English conversational telephone speech recognition by humans and machines. In *Interspeech*, 2017.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Tijmen Tieleman and Geoffrey Hinton. Lecture 6.5-rmsprop, coursera: Neural networks for machine learning. *University of Toronto, Technical Report*, 2012.

Mengdi Wang, Ethan X Fang, and Han Liu. Stochastic compositional gradient descent: algorithms for minimizing compositions of expected-value functions. *Mathematical Programming*, 161(1-2): 419–449, 2017.

Zhe Wang, Kaiyi Ji, Yi Zhou, Yingbin Liang, and Vahid Tarokh. SpiderBoost and momentum: Faster variance reduction algorithms. In *Advances in Neural Information Processing Systems*, pp. 2406–2416, 2019.

Rachel Ward, Xiaoxia Wu, and Leon Bottou. AdaGrad stepsizes: Sharp convergence over nonconvex landscapes. In *International Conference on Machine Learning*, pp. 6677–6686, 2019.

Ashia C Wilson, Rebecca Roelofs, Mitchell Stern, Nati Srebro, and Benjamin Recht. The marginal value of adaptive gradient methods in machine learning. In *Advances in Neural Information Processing Systems*, pp. 4148–4158, 2017.

Yang You, Igor Gitman, and Boris Ginsburg. Scaling SGD batch size to 32k for imagenet training. *arXiv preprint arXiv:1708.03888*, 6, 2017.

Yang You, Jing Li, Sashank Reddi, Jonathan Hseu, Sanjiv Kumar, Srinadh Bhojanapalli, Xiaodan Song, James Demmel, Kurt Keutzer, and Cho-Jui Hsieh. Large batch optimization for deep learning: Training BERT in 76 minutes. *arXiv preprint arXiv:1904.00962*, 2019.

Wei Zhang, Xiaodong Cui, Ulrich Finkler, Brian Kingsbury, George Saon, David Kung, and Michael Picheny. Distributed deep learning strategies for automatic speech recognition. In *ICASSP'2019*, May 2019.

Dongruo Zhou, Pan Xu, and Quanquan Gu. Stochastic nested variance reduction for nonconvex optimization. In *Advances in Neural Information Processing Systems*, pp. 3921–3932, 2018.