SUPER-ADAM: Faster and Universal Framework of Adaptive Gradients

Feihu Huang, Junyi Li, Heng Huang

Department of Electrical and Computer Engineering, University of Pittsburgh, USA

NeurIPS | 2021

Thirty-fifth Conference on Neural Information Processing Systems

Outline

- Background
- Existing Adaptive Gradient Methods
- Our Super-Adam Algorithm
- Convergence Analysis
- Experimental Results
- Conclusions

Outline

- Background
- Existing Adaptive Gradient Methods
- Our Super-Adam Algorithm
- Convergence Analysis
- Experimental Results
- Conclusions

Background

In many machine learning applications, to obtain an intelligent agent, we frequently need large models and big data.

For example, in image classification, we usually meet big datasets such as ImageNet including more than 14 million labeled images, and use these big data to train large deep neural networks (DNNs).

Background

Recently, Stochastic Gradient Descent (SGD) is a workhorse in solving these large-scale machine learning problems, due to only requiring a mini-batch samples or even one sample at each iteration.

Adaptive gradient methods are one of the most important variants of SGD, which use adaptive learning rates and possibly incorporate momentum techniques, so they generally require little parameter tuning and enjoy faster convergence rate than SGD.

Outline

- Background
- Existing Adaptive Gradient Methods
- Our Super-Adam Algorithm
- Convergence Analysis
- Experimental Results
- Conclusions

Existing Adaptive Gradient Methods

In the paper, we consider solving the following stochastic optimization problem:

$$\min_{x\in\mathcal{X}} f(x) := \mathbb{E}_{\xi\sim\mathcal{D}}[f(x;\xi)] \tag{1}$$

where $f(x)$ is a smooth and possibly nonconvex function, and ξ is a random variable following an unknown distribution $\mathcal{D}$. Here $\mathcal{X} \subseteq \mathbb{R}^d$ is a compact and convex set. The problem (1) is frequently appears in many machine learning applications such as the expectation loss minimization.

In the subsection, we review the existing typical adaptive gradient methods. Recently, many adaptive algorithms have been proposed to solve the problem (1), and achieve good performances. For example, Adagrad [12] is the first adaptive gradient method with adaptive learning rate for each individual dimension. Specifically, it adopts the following update form:

$$x_{t+1} = x_t - \eta_t \frac{g_t}{\sqrt{v_t}}, \quad v_t = \frac{1}{t}\sum_{j=1}^{t} g_j^2, \tag{2}$$

where $g_t = \nabla f(x_t;\xi_t)$ is the stochastic gradient, and $\eta_t = \frac{\eta}{\sqrt{t}}$ with $\eta > 0$ is the step size. In fact, η_t only is the basic learning rate that is the same for all coordinates of variable x_t, while $\frac{\eta_t}{\sqrt{v_{t,i}}}$ is the effective learning rate for the i-th coordinate of x_t, which changes across the coordinates.

Existing Adaptive Gradient Methods

Adam [20] is one of the most popular exponential moving average variant of Adagrad, which combines the exponential moving average technique with momentum acceleration. Its update form:

$$m_t = \alpha_1 m_{t-1} + (1-\alpha_1)\nabla f(x_t;\xi_t), \quad v_t = \alpha_2 v_{t-1} + (1-\alpha_2)(\nabla f(x_t;\xi_t))^2$$
$$\hat{m}_t = m_t/(1-\alpha_1^t), \quad \hat{v}_t = v_t/(1-\alpha_2^t), \quad x_{t+1} = x_t - \eta_t \hat{m}_t/(\sqrt{\hat{v}_t}+\varepsilon), \tag{3}$$

where $\alpha_1 > 0$, $\alpha_2 > 0$ and $\varepsilon > 0$, and $\eta_t = \frac{\eta}{\sqrt{t}}$ with $\eta > 0$. However, Reddi et al. [25] found a divergence issue of the Adam algorithm, and proposed a modified version of Adam (i.e., Amsgrad), which adopts a new step instead of the debiasing step in (3) to ensure the decay of the effective learning rate:

$$\hat{v}_t = \max(\hat{v}_{t-1}, v_t), \quad x_{t+1} = x_t - \eta_t m_t/\sqrt{\hat{v}_t}. \tag{4}$$

Existing Adaptive Gradient Methods

Due to using the coordinate-wise learning rates, these adaptive gradient methods frequently have worse generalization performance than SGD (with momentum) [31]. To improve the performance, AdamW [24] uses a decoupled weight decay regularization, defined as

$$g_t = \nabla f(x_t;\xi_t) + \lambda x_t, \quad m_t = \alpha_1 m_{t-1} + (1-\alpha_1) g_t, \quad v_t = \alpha_2 v_{t-1} + (1-\alpha_2) g_t^2$$
$$\hat{m}_t = m_t/(1-\alpha_1^t), \quad \hat{v}_t = v_t/(1-\alpha_2^t), \quad x_{t+1} = x_t - \eta_t\big(\alpha \hat{m}_t/(\sqrt{\hat{v}_t}+\varepsilon) + \lambda x_t\big), \tag{5}$$

where $\alpha > 0$ and $\lambda > 0$. More recently, to improve the performance, AdaBelief [36] adopts a stepsize according to the 'belief' in the current gradient direction, defined as

$$m_t = \alpha_1 m_{t-1} + (1-\alpha_1)\nabla f(x_t;\xi_t), \quad v_t = \alpha_2 v_{t-1} + (1-\alpha_2)(\nabla f(x_t;\xi_t) - m_t)^2 + \varepsilon$$
$$\hat{m}_t = m_t/(1-\alpha_1^t), \quad \hat{v}_t = v_t/(1-\alpha_2^t), \quad x_{t+1} = x_t - \eta_t \hat{m}_t/(\sqrt{\hat{v}_t}+\varepsilon), \quad \forall\, t \geq 1 \tag{6}$$

where $\alpha_1 > 0, \alpha_2 > 0$, and $\eta_t = \frac{\eta}{\sqrt{t}}$ with $\eta > 0$, and $\varepsilon > 0$.

Existing Adaptive Gradient Methods

So far, these adaptive gradient methods use the coordinate-wise learning rate.

At the same time, to improve the generalization performance, recently some effective adaptive gradient methods [30, 21, 11] were proposed with adopting the global adaptive learning rates instead of coordinate-wise counterparts. For example, AdaGrad-Norm [30] applies a global adaptive learning rate to the following update form for all $t \geq 1$

$$x_t = x_{t-1} - \eta \nabla f(x_{t-1}; \xi_{t-1})/b_t, \quad b_t^2 = b_{t-1}^2 + \|\nabla f(x_{t-1}; \xi_{t-1})\|^2, \ b_0 > 0, \tag{7}$$

where $\eta > 0$. The adaptive-SGD [21] adopts the global adaptive learning rate, defined as

$$\eta_t = \frac{k}{\left(\omega + \sum_{i=1}^{t-1} \|\nabla f(x_i; \xi_i)\|^2\right)^{1/2+\varepsilon}}, \quad x_{t+1} = x_t - \eta_t \nabla f(x_t; \xi_t), \tag{8}$$

where $k > 0$, $\omega > 0$, and $\varepsilon \geq 0$. Subsequently, STORM [11] not only uses a global adaptive learning rate but also adopts the variance-reduced technique in gradient estimator to accelerate algorithm, defined as

$$\eta_t = \frac{k}{\left(\omega + \sum_{i=1}^{t} \|\nabla f(x_i; \xi_i)\|^2\right)^{1/3}}, \quad x_{t+1} = x_t - \eta_t g_t, \tag{9}$$
$$g_{t+1} = \nabla f(x_{t+1}; \xi_{t+1}) + (1 - c\eta_t^2)(g_t - \nabla f(x_t; \xi_{t+1})),$$

where $k > 0$, $\omega > 0$ and $c > 0$.

Outline

- Background
- Existing Adaptive Gradient Methods
- Our Super-Adam Algorithm
- Convergence Analysis
- Experimental Results
- Conclusions

Super-Adam Algorithm

So far, the existing adaptive gradient methods only focus on some specific adaptive learning rates. It is desired to design a universal framework for practical algorithms of adaptive gradients with theoretical guarantee.

Insight from the dynamic mirror descent algorithm, we propose an efficient adaptive gradient framework, i.e., Super-Adam Algorithm, based on the momentum techniques.

Dynamic Mirror Descent iteration: $\min_{x\in\mathcal{C}} f(x)$

$$x^{t+1} = \arg\min_{x\in\mathcal{C}} \left\{ f(x^t) + \langle \nabla f(x^t), x - x^t \rangle + \frac{1}{\eta_t} D_{\varphi_t}(x, x^t) \right\}$$

Bregman Distance: $D_{\varphi_t}(x, x^t) = \varphi_t(x) - \varphi_t(x^t) - \langle \nabla \varphi_t(x^t), x - x^t \rangle$

Bregman function varies with the iteration.

Super-Adam Algorithm

Algorithm 1 SUPER-ADAM Algorithm

Input: Total iteration T, and tuning parameters $\{\mu_t, \alpha_t\}_{t=1}^T$, $\gamma > 0$;

Initialize: $v_1 = 0$, and $x_1 \in \mathcal{X}$, sample one point ξ_1, compute $g_1 = \nabla f(x_1; \xi_1)$;

for $t = 1, 2, \ldots, T$ **do**

  Generate the adaptive matrix $H_t \in \mathbb{R}^{d\times d}$; // Given two examples to update H_t:

  Case 1: given $\beta \in (0, 1)$ and $\lambda > 0$ to update H_t,

  $v_t = \beta v_{t-1} + (1-\beta)\nabla f(x_t; \xi_t)^2$, $H_t = \text{diag}(\sqrt{v_t} + \lambda)$;

  Case 2: given $\beta \in (0, 1)$ and $\lambda > 0$ to update H_t,

  $b_t = \beta b_{t-1} + (1-\beta)\|\nabla f(x_t; \xi_t)\|$, $H_t = (b_t + \lambda) I_d$;

  Update $\tilde{x}_{t+1} = \arg\min_{x\in\mathcal{X}} \{\langle g_t, x\rangle + \frac{1}{2\gamma}(x - x_t)^T H_t (x - x_t)\}$;

  Update $x_{t+1} = (1-\mu_t)x_t + \mu_t \tilde{x}_{t+1}$;

  Sample one point ξ_{t+1}, and compute $g_{t+1} = \alpha_{t+1}\nabla f(x_{t+1}; \xi_{t+1}) + (1-\alpha_{t+1})\big[g_t + \tau\big(\nabla f(x_{t+1}; \xi_{t+1}) - \nabla f(x_t; \xi_{t+1})\big)\big]$, where $\tau \in \{0, 1\}$;

end for

Output: (for theoretical) x_ζ chosen uniformly random from $\{x_t\}_{t=1}^T$; (for practical) x_T.

Super-Adam Algorithm

Let $w_t(x) = \frac{1}{2}x^T H_t x$, we give a prox-function (i.e., Bregman distance) [4, 5, 14] associated with $w_t(x)$, defined as:

$$V_t(x, x_t) = w_t(x) - \left[w_t(x_t) + \langle \nabla w_t(x_t), x - x_t \rangle\right] = \frac{1}{2}(x - x_t)^T H_t (x - x_t). \tag{16}$$

Thus, the step 9 of Algorithm 1 is equivalent to the following generalized projection problem:

$$\tilde{x}_{t+1} = \arg\min_{x \in \mathcal{X}} \left\{ \langle g_t, x \rangle + \frac{1}{\gamma} V_t(x, x_t) \right\}, \tag{17}$$

where $\gamma > 0$. In fact, solving the above subproblem (17) can be seen as a mirror descent iteration [5, 3]. As in [14], we define a gradient mapping $\mathcal{G}_{\mathcal{X}}(x_t, \nabla f(x_t), \gamma) = \frac{1}{\gamma}(x_t - x_{t+1}^+)$, where

$$x_{t+1}^+ = \arg\min_{x \in \mathcal{X}} \left\{ \langle \nabla f(x_t), x \rangle + \frac{1}{\gamma} V_t(x, x_t) \right\}. \tag{18}$$

Super-Adam Algorithm

At the step 11 of Algorithm 1, we use the stochastic gradient estimator g_{t+1} for all $t \geq 1$:

$$g_{t+1} = \alpha_{t+1}\nabla f(x_{t+1};\xi_{t+1}) + (1-\alpha_{t+1})\big[g_t + \tau\big(\nabla f(x_{t+1};\xi_{t+1}) - \nabla f(x_t;\xi_{t+1})\big)\big], \quad (14)$$

where $\tau \in \{0,1\}$ and $\alpha_{t+1} \in (0,1]$ for all $t \geq 1$. When $\tau = 1$, we have $g_{t+1} = \nabla f(x_{t+1};\xi_{t+1}) + (1-\alpha_{t+1})\big(g_t - \nabla f(x_t;\xi_{t+1})\big)$ for all $t \geq 1$, which is a momentum-based variance reduced gradient estimator used in STORM [11]. When $\tau = 0$, we have $g_{t+1} = \alpha_{t+1}\nabla f(x_{t+1};\xi_{t+1}) + (1-\alpha_{t+1})g_t$ for all $t \geq 1$, which is a basic momentum gradient estimator used in the Adam [20].

Outline

- Background
- Existing Adaptive Gradient Methods
- Our Super-Adam Algorithm
- Convergence Analysis
- Experimental Results
- Conclusions

Convergence Analysis

4.1 Some Mild Assumptions

Assumption 1. *Variance of unbiased stochastic gradient is bounded,* i.e., *there exists a constant* $\sigma > 0$ *such that for all* $x \in \mathcal{X}$, *it follows* $\mathbb{E}[\nabla f(x;\xi)] = \nabla f(x)$ *and* $\mathbb{E}\|\nabla f(x;\xi) - \nabla f(x)\|^2 \leq \sigma^2$.

Assumption 2. *The function* $f(x)$ *is bounded from below in* $\mathcal{X}$, i.e., $f^* = \inf_{x\in\mathcal{X}} f(x)$.

Assumption 3. *Assume the adaptive matrix* H_t *for all* $t \geq 1$ *satisfies* $H_t \succeq \rho I_d \succ 0$, *and* $\rho > 0$ *denotes a lower bound of the smallest eigenvalue of* H_t *for all* $t \geq 1$.

Assumption 1 is commonly used in stochastic optimization [12, 9]. Assumption 2 ensures the feasibility of the problem (1). In fact, all adaptive algorithms in Table 1 require these mild Assumptions 1 and 2. Assumption 3 guarantees that the adaptive matrices $\{H_t\}_{t\geq 1}$ used in our algorithm are positive definite and their smallest eigenvalues have a lower bound $\rho > 0$. From the above adaptive

Convergence Analysis

Assumption 1 is commonly used in stochastic optimization [14, 11]. Assumption 2 ensures the feasibility of the problem (1). In fact, all adaptive algorithms in Table 1 require these mild Assumptions 1 and 2. Assumption 3 guarantees that the adaptive matrices $\{H_t\}_{t\geq 1}$ used in our algorithm are positive definite and their smallest eigenvalues have a lower bound $\rho > 0$. From the above adaptive matrices used in our algorithm, we have $\rho \geq \lambda > 0$. In fact, many adaptive algorithms also implicitly use Assumption 3. For example, Zaheer et al. [33] and Zhuang et al. [37] used the following iteration form to update the variable x: $x_{t+1} = x_t - \eta_t \frac{m_t}{\sqrt{v_t}+\varepsilon}$ for all $t \geq 0$ and $\varepsilon > 0$, which is equivalent to $x_{t+1} = x_t - \eta_t H_t^{-1} m_t$ with $H_t = \mathrm{diag}(\sqrt{v_t} + \varepsilon)$. Clearly, we have $H_t \succeq \varepsilon I_d \succ 0$. Ward et al. [31] applied a global adaptive learning rate to the update form in (7), which is equivalent to the following form: $x_t = x_{t-1} - \eta H_t^{-1} \nabla f(x_{t-1}; \xi_{t-1})$ with $H_t = b_t I_d$. By the above (7), we have $H_t \succeq \cdots \succeq H_0 = b_0 I_d \succ 0$. Li et al. [22] and Cutkosky et al. [11] applied a global adaptive learning rate to the update forms in (8) and (9), which is equivalent to $x_{t+1} = x_t - H_t^{-1} g_t$, where $H_t = (1/\eta_t) I_d$ and $\eta_t = k/\big(\omega + \sum_{i=1}^{t} \|\nabla f(x_i; \xi_i)\|^2\big)^{\alpha}$ with $k > 0, \omega > 0, \alpha \in (0, 1)$. By the above (8) and (9), we have $H_t \succeq \cdots \succeq H_0 = (\omega^{\alpha}/k) I_d \succ 0$. Reddi et al. [26] and Chen et al. [6] used the condition $\hat{v}_t = \max(\hat{v}_{t-1}, v_t)$, and let $H_t = \mathrm{diag}(\sqrt{\hat{v}_t})$, thus we have $H_t \succeq \cdots \succeq H_1 = \mathrm{diag}(\sqrt{\hat{v}_1}) = \sqrt{1-\alpha_2}\,\mathrm{diag}(|\nabla f(x_1; \xi_1)|) \succeq 0$. Without loss of generality, choosing an initial point x_1 and let $(\nabla f(x_1; \xi_1))_j \neq 0$ for all $j \in [d]$, we have $H_t \succeq \cdots \succeq H_1 \succ 0$. When $\mathcal{X} = \mathbb{R}^d$ and $H_t \neq I_d$, interestingly, our SUPER-ADAM algorithm also includes a class of novel momentum-based quasi-Newton algorithms by choosing an approximated Hessian matrix H_t. In fact, the quasi-Newton algorithms [30, 15, 35] generally require the bounded approximated Hessian matrices, i.e., $\bar{\kappa} \leq \lambda_{\min}(H_t) \leq \lambda_{\max}(H_t) \leq \hat{\kappa}$ for all $t \geq 1$, where $\hat{\kappa} \geq \bar{\kappa} > 0$. Thus the above Assumption 3 is reasonable and mild. Note that we mainly focus on the adaptive algorithms in this paper, while the quasi-Newton algorithms only are the by-products of our algorithm.

Convergence Analysis

4.3 Convergence Analysis of SUPER-ADAM ($\tau = 1$)

In this subsection, we provide the convergence analysis of our SUPER-ADAM ($\tau = 1$) algorithm when using the momentum-based variance reduced gradient estimator [9, 24]. The detail proofs are given in the Appendix A.1.

Assumption 4. *Each component function $f(x;\xi)$ is L-smooth for all $\xi \in \mathcal{D}$, such that*

$$\|\nabla f(x;\xi) - \nabla f(y;\xi)\| \leq L\|x - y\|, \ \forall x, y \in \mathcal{X}.$$

Assumption 4 is widely used in the variance-reduced algorithms [11, 9]. According to Assumption 4, we have $\|\nabla f(x) - \nabla f(y)\| = \|\mathbb{E}[\nabla f(x;\xi) - \nabla f(y;\xi)]\| \leq \mathbb{E}\|\nabla f(x;\xi) - \nabla f(y;\xi)\| \leq L\|x - y\|$ for all $x, y \in \mathcal{X}$. So the function $f(x)$ is L-smooth.

Convergence Analysis

Theorem 1. *In Algorithm 1, under the Assumptions (**1,2,3,4**), given* $\tau = 1$, $\mu_t = \frac{k}{(m+t)^{1/3}}$ *and* $\alpha_{t+1} = c\mu_t^2$ *for all* $t \geq 0$, $0 < \gamma \leq \frac{\rho m^{1/3}}{4kL}$, $\frac{1}{k^3} + \frac{10L^2\gamma^2}{\rho^2} \leq c \leq \frac{m^{2/3}}{k^2}$, $m \geq \max\left(\frac{3}{2}, k^3, (ck)^3, \frac{8^{3/2}}{(3k)^{3/2}}\right)$ *and* $k > 0$, *we have*

$$\frac{1}{T}\sum_{t=1}^{T}\mathbb{E}\|\mathcal{G}_{\mathcal{X}}(x_t, \nabla f(x_t), \gamma)\| \leq \frac{1}{T}\sum_{t=1}^{T}\mathbb{E}\big[\mathcal{M}_t\big] \leq \frac{2\sqrt{2G}m^{1/6}}{T^{1/2}} + \frac{2\sqrt{2G}}{T^{1/3}}, \tag{18}$$

where $G = \frac{f(x_1)-f^*}{k\rho\gamma} + \frac{m^{1/3}\sigma^2}{8k^2L^2\gamma^2} + \frac{k^2c^2\sigma^2}{4L^2\gamma^2}\ln(m+T)$.

Remark 1. *Without loss of generality, let* $\rho = O(1)$, $\gamma = O(1)$, $k = O(1)$ *and* $m = O(1)$, *we have* $G = O\big(\ln(m+T)\big) = \tilde{O}(1)$. *Thus, our SUPER-ADAM* ($\tau = 1$) *algorithm has convergence rate of* $\tilde{O}\big(\frac{1}{T^{1/3}}\big)$. *Consider* $\frac{1}{T^{1/3}} \leq \epsilon$, *we have* $T \geq \epsilon^{-3}$. *Since our algorithm requires one sample to estimate the stochastic gradient* g_t *at each iteration, and needs* T *iterations. Thus, our SUPER-ADAM* ($\tau = 1$) *algorithm has a sample complexity of* $1 \cdot T = \tilde{O}(\epsilon^{-3})$ *for finding an* ϵ*-stationary point of the problem (1).*

Convergence Analysis

Corollary 1. *In Algorithm 1 under the above Assumptions (**1,2,3,4**), let* $\mathcal{X} = \mathbb{R}^d$, *and given* $\tau = 1$, $\mu_t = \frac{k}{(m+t)^{1/3}}$ *and* $\alpha_{t+1} = c\mu_t^2$ *for all* $t \geq 0$, $\gamma = \frac{\rho m^{1/3}}{4kL}$, $\frac{1}{k^3} + \frac{10L^2\gamma^2}{\rho^2} \leq c \leq \frac{m^{2/3}}{k^2}$, $m \geq \max\left(\frac{3}{2}, (ck)^3, k^3, \frac{8^{3/2}}{(3k)^{3/2}}\right)$ *and* $k > 0$, *we have*

$$\frac{1}{T}\sum_{t=1}^{T}\mathbb{E}\|\nabla f(x_t)\| \leq \frac{\max_{1\leq t\leq T}\|H_t\|}{\rho}\left(\frac{2\sqrt{2G'}}{T^{1/2}} + \frac{2\sqrt{2G'}}{m^{1/6}T^{1/3}}\right), \tag{20}$$

where $G' = 4L(f(x_1) - f^*) + 2\sigma^2 + \frac{4k^4c^2\sigma^2}{m^{1/3}}\ln(m+T)$.

Remark 2. *Under the same conditions in Theorem 1, based on the standard metric* $\mathbb{E}\|\nabla f(x_t)\|$ *the our SUPER-ADAM* ($\tau = 1$) *algorithm still has a sample complexity of* $\tilde{O}(\epsilon^{-3})$ *for finding an* ϵ*-stationary point of the problem (1) with* $\mathcal{X} = \mathbb{R}^d$. *Interestingly, the right of the above inequality (20) includes a term* $\frac{\max_{1\leq t\leq T}\|H_t\|}{\rho}$ *that can be seen as an upper bound of the condition number of adaptive matrices* $\{H_t\}_{t=1}^T$. *In fact, when using* H_t *given in the above **case 1**, we have* $\frac{\max_{1\leq t\leq T}\|H_t\|}{\rho} \leq \frac{G_1+\lambda}{\lambda}$ *as in the existing adaptive gradient methods assuming the bounded stochastic gradient* $\|\nabla f(x;\xi)\|_\infty \leq G_1$; *When using* H_t *given in the above **case 2**, we have* $\frac{\max_{1\leq t\leq T}\|H_t\|}{\rho} \leq \frac{G_2+\sigma+\lambda}{\lambda}$ *as in the existing adaptive gradient methods assuming the bounded full gradient* $\|\nabla f(x)\| \leq G_2$; *When using* H_t *given in the above **case 3**, we have* $\frac{\max_{1\leq t\leq T}\|H_t\|}{\rho} \leq \frac{L+\lambda}{\lambda}$. *When using* H_t *given in the above **case 4**, we have* $\frac{\max_{1\leq t\leq T}\|H_t\|}{\rho} \leq \frac{2G_1+\lambda}{\lambda}$. ***Note that** we can't take it for granted that these parameters* ρ *and* m *can clearly affects convergence rate since* $\frac{1}{k^3} + \frac{10L^2\gamma^2}{\rho^2} \leq c \leq \frac{m^{2/3}}{k^2}$, $\gamma = \frac{\rho m^{1/3}}{4kL}$ *and* G' *includes the parameters* c *and* m.

Convergence Analysis

4.4 Convergence Analysis of SUPER-ADAM ($\tau = 0$)

In this subsection, we provide the convergence analysis of our SUPER-ADAM ($\tau = 0$) algorithm when using the basic momentum stochastic gradient estimator [16]. The detail proofs are given in the Appendix A.2.

Assumption 5. *The function $f(x) = \mathbb{E}_\xi[f(x;\xi)]$ is L-smooth such that*

$$\|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\|, \ \forall x, y \in \mathcal{X}.$$

Assumption 5 is widely used in the adaptive algorithms [28, 6, 31]. Clearly, Assumption 5 is milder than Assumption 4.

Theorem 2. *In Algorithm 1, under the Assumptions (**1,2,3,5**), given $\tau = 0$, $\mu_t = \frac{k}{(m+t)^{1/2}}$, $\alpha_{t+1} = c\mu_t$ for all $t \geq 0$, $0 < \gamma \leq \frac{\rho m^{1/2}}{8Lk}$, $\frac{8L\gamma}{\rho} \leq c \leq \frac{m^{1/2}}{k}$, $m \geq \max\{k^2, (ck)^2\}$ and $k > 0$, we have*

$$\frac{1}{T}\sum_{t=1}^{T}\mathbb{E}\|\mathcal{G}_{\mathcal{X}}(x_t, \nabla f(x_t), \gamma)\| \leq \frac{1}{T}\sum_{t=1}^{T}\mathbb{E}[\mathcal{M}_t] \leq \frac{2\sqrt{2M}m^{1/4}}{T^{1/2}} + \frac{2\sqrt{2M}}{T^{1/4}},$$

where $M = \frac{f(x_1)-f^*}{\rho\gamma k} + \frac{2\sigma^2}{\rho\gamma kL} + \frac{2m\sigma^2}{\rho\gamma kL}\ln(m+T)$.

Convergence Analysis

Remark 3. *Without loss of generality, let* $\rho = O(1)$, $k = O(1)$, $m = O(1)$ *and* $\gamma = O(1)$, *we have* $M = O\big(\ln(m+T)\big) = \tilde{O}(1)$. *Thus, our SUPER-ADAM* ($\tau = 0$) *algorithm has convergence rate of* $\tilde{O}\big(\frac{1}{T^{1/4}}\big)$. *Consider* $\frac{1}{T^{1/4}} \leq \epsilon$, *we have* $T \geq \epsilon^{-4}$. *Since our algorithm requires one sample to estimate the stochastic gradient* g_t *at each iteration, and needs* T *iterations. Thus, our SUPER-ADAM* ($\tau = 0$) *algorithm has a sample complexity of* $1 \cdot T = \tilde{O}(\epsilon^{-4})$ *for finding an* ϵ*-stationary point of the problem* (1).

Corollary 2. *In Algorithm 1, under the above Assumptions (**1,2,3,5**), let* $\mathcal{X} = \mathbb{R}^d$, *and given* $\tau = 0$, $\mu_t = \frac{k}{(m+t)^{1/2}}$, $\alpha_{t+1} = c\mu_t$ *for all* $t \geq 0$, $k > 0$, $\gamma = \frac{\rho m^{1/2}}{8Lk}$, $\frac{8L\gamma}{\rho} \leq c \leq \frac{m^{1/2}}{k}$, *and* $m \geq \max\{k^2, (ck)^2\}$, *we have*

$$\frac{1}{T}\sum_{t=1}^{T}\mathbb{E}\|\nabla f(x_t)\| \leq \frac{\max_{1\leq t\leq T}\|H_t\|}{\rho}\left(\frac{2\sqrt{2M'}}{T^{1/2}} + \frac{2\sqrt{2M'}}{m^{1/4}T^{1/4}}\right),$$

where $M' = 8L(f(x_1) - f^*) + 16\sigma^2 + 16m\sigma^2\ln(m+T)$.

Remark 4. *Under the same conditions in Theorem 1, based on the standard metric* $\mathbb{E}\|\nabla f(x_t)\|$ *the our SUPER-ADAM* ($\tau = 0$) *algorithm still has a sample complexity of* $\tilde{O}(\epsilon^{-4})$ *for finding an* ϵ*-stationary point of the problem* (1) *with* $\mathcal{X} = \mathbb{R}^d$.

Convergence Analysis

Table 1: Convergence properties of the representative adaptive gradient algorithms for finding an ϵ-stationary point of the **non-convex** stochastic problem (1), i.e., $\mathbb{E}\|\nabla f(x)\| \leq \epsilon$ or its equivalent variants. For fair comparison, we only give the gradient complexity and convergence rate in **the worst case** without considering the sparsity of stochastic gradient. Here T denotes the whole number of iteration, b denotes mini-batch size, and d denotes dimension of data. **ALR** denotes adaptive learning rate. **1** denotes the smoothness of each component function $f(x;\xi)$; **2** denotes the smoothness of objective function $f(x) = \mathbb{E}_{\xi}[f(x;\xi)]$; **3** denotes the bounded noisy gradient $\nabla f(x;\xi)$; **4** denotes the bounded true gradient $\nabla f(x)$; **5** denotes that $f(x)$ is Lipschitz continuous; **6** denotes the smoothness of true gradient $\nabla f(x)$.

Algorithm	Reference	Complexity	Convergence Rate	ALR	Conditions
Adam/ YOGI	[28]	$O(\epsilon^{-4})$	$O(\frac{1}{\sqrt{T}} + \frac{1}{\sqrt{b}})$	specific	**1, 2, 3, 4**
Generalized Adam	[6]	$\tilde{O}(\epsilon^{-4})$	$O(\frac{\sqrt{\log(T)}}{T^{1/4}})$	specific	**2, 3, 4**
Padam	[5]	$O(\epsilon^{-4})$	$O(\frac{1}{\sqrt{T}} + \frac{1}{T^{1/4}})$	specific	**2, 3, 4**
Adaptive SGD	[17]	$\tilde{O}(\epsilon^{-4})$	$O(\frac{\sqrt{\ln(T)}}{\sqrt{T}} + \frac{\sqrt{\ln(T)}}{T^{1/4}})$	specific	**2, 5**
AdaGrad-Norm	[26]	$\tilde{O}(\epsilon^{-4})$	$O(\frac{\sqrt{\log(T)}}{T^{1/4}})$	specific	**2, 4**
Ada-Norm-SGD	[8]	$\tilde{O}(\epsilon^{-3.5})$	$\tilde{O}(\frac{1}{T^{2/7}})$	specific	**2, 6**
AdaBelief	[31]	$\tilde{O}(\epsilon^{-4})$	$O(\frac{\sqrt{\log(T)}}{T^{1/4}})$	specific	**2, 3, 4**
Adam$^+$	[19]	$O(\epsilon^{-3.5})$	$O(\frac{1}{T^{2/7}})$	specific	**2, 6**
STORM	[9]	$\tilde{O}(\epsilon^{-3})$	$O(\frac{(\ln(T))^{3/4}}{\sqrt{T}} + \frac{\sqrt{\ln(T)}}{T^{1/3}})$	specific	**1, 3, 4**
SUPER-ADAM ($\tau = 0$)	Ours	$\tilde{O}(\epsilon^{-4})$	$O(\frac{\sqrt{\ln(T)}}{\sqrt{T}} + \frac{\sqrt{\ln(T)}}{T^{1/4}})$	universal	**2**
SUPER-ADAM ($\tau = 1$)	Ours	$\tilde{O}(\epsilon^{-3})$	$O(\frac{\sqrt{\ln(T)}}{\sqrt{T}} + \frac{\sqrt{\ln(T)}}{T^{1/3}})$	universal	**1**

Outline

- Background
- Existing Adaptive Gradient Methods
- Our Super-Adam Algorithm
- Convergence Analysis
- Experimental Results
- Conclusions

Experimental Results

Table 2: Summary of setups in the experiments.

Task	*Architecture*	*Dataset*
Image Classification	ResNet18	CIFAR-10
Image Classification	VGG19	CIFAR-100
Image Classification	ResNet34	Image-Net
Language Modeling	Two-layer LSTM	Wiki-Text2
Language Modeling	Transformer	Wiki-Text2

Experimental Results

6.1 Image Classification Task

In the experiment, we conduct image classification task on CIFAR-10, CIFAR-100 and Image-Net datasets. We show results for CIFAR-10 and CIFAR-100 in the main-text and for Image-net in the Appendix B due to space limitation. Specifically, we perform training over ResNet-18 [12] and VGG-19 [20] on CIFAR-10 and CIFAR-100 datasets. For all the optimizers, we set the batch size as 128 and trains for 200 epochs. For the learning rates and other hyper-parameters, we do grid search and report the best one for each optimizer. In Adam, Amsgrad and AdaBelief algorithms, we set the learning rate as 0.001. In AdaGrad-Norm, the best learning rate is 17 for CIFAR-10 and 10 for CIFAR-100, respectively. In Adam$^+$, we use the recommended tuning parameters in [17]. In STORM, the best result is obtained when $w = 6$, $k = 10$ and $c = 100$ for CIFAR-10, while $w = 3$, $k = 10$ and $c = 100$ for CIFAR-100. For our SUPER-ADAM algorithm. For both CIFAR-10 and CIFAR-100 data-set, we set $k = 1$, $m = 100$, $c = 40$, $\gamma = 0.001$ when $\tau = 1$, and $k = 1$, $m = 100$, $c = 20$, $\gamma = 0.001$ when $\tau = 0$.

Experimental Results

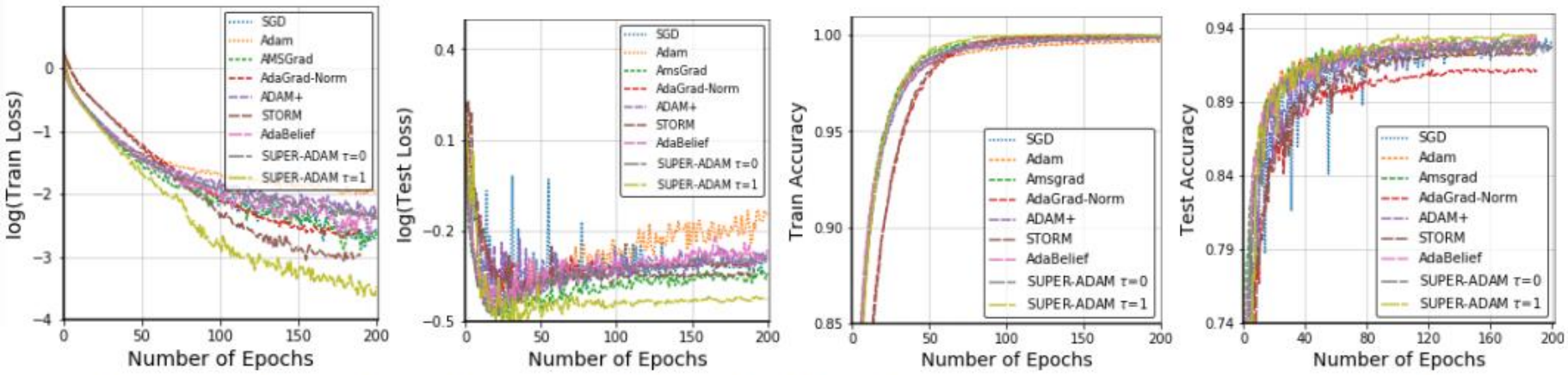

Figure 1: Experimental Results of CIFAR-10 by Different Optimizers over ResNet-18.

Experimental Results

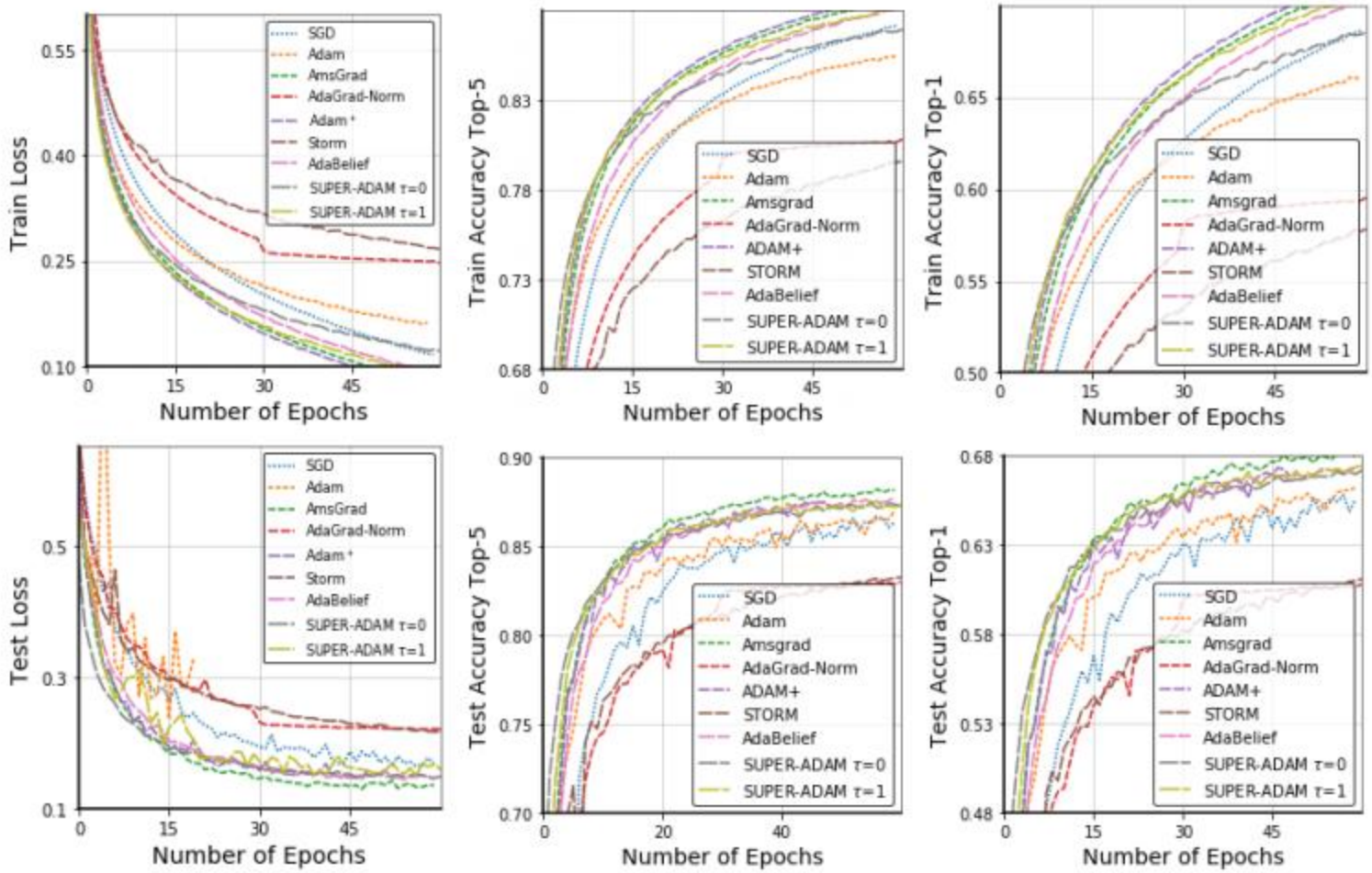

Figure 5: Experimental Results of Image-Net by Different Optimizers over ResNet-34.

Experimental Results

6.2 Language Modeling Task

In the experiment, we conduct language modeling task on the Wiki-Text2 dataset. Specifically, we train a 2-layer LSTM [13] and a 2-layer Transformer over the WiKi-Text2 dataset. We present the results trained with LSTM here and defer the results for Transformer to the Appendix B. For the LSTM, we use 650 dimensional word embeddings and 650 hidden units per-layer. What's more, we set the batch size as 20 and trains for 40 epochs with dropout rate 0.5. We also clip the gradients by norm 0.25 in case of the exploding gradient in LSTM. We also decrease the learning by 4 whenever the validation error increases. For the learning rate, we also do grid search and report the best one for each optimizer. In Adam and Amsgrad algorithms, we set the learning rate as 0.001 in LSTM In AdaGrad-Norm algorithm, the best learning rate is 40. In Adam$^+$ algorithm, we use the learning rate 20. In AdaBelief algorithm, we set the learing rate 0.1. In STORM algorithm, we set $w = 50$, $k = 10$ and $c = 100$. In our SUPER-ADAM algorithm, we set $k = 1$, $m = 100$, $c = 40$, $\gamma = 0.001$ when $\tau = 1$, while $k = 1$, $m = 100$, $c = 20$, $\gamma = 0.01$ when $\tau = 0$.

Experimental Results

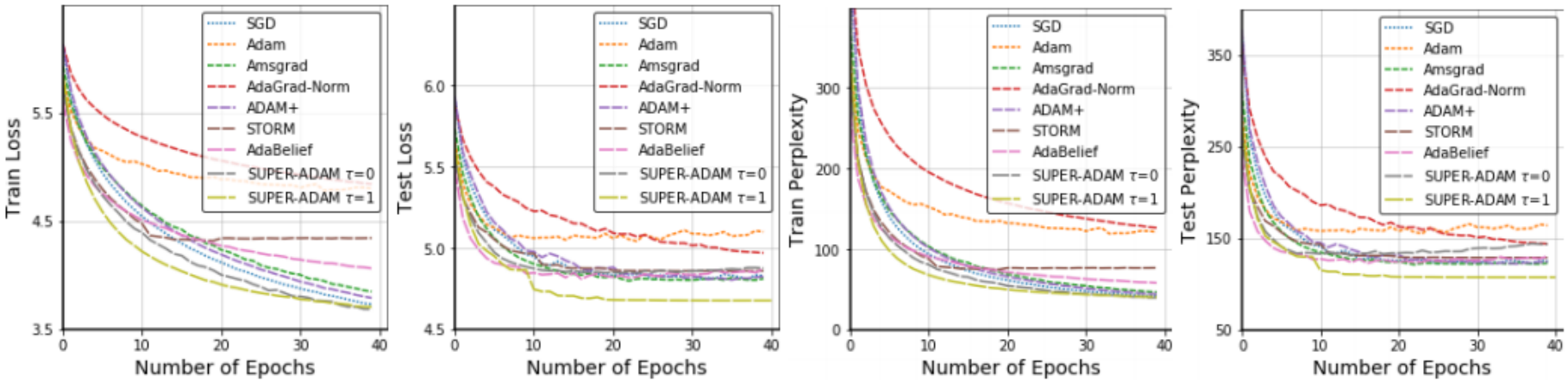

Figure 6: Experimental Results of WikiText-2 by Different Optimizers over LSTM.

Outline

- Background
- Existing Adaptive Gradient Methods
- Our Super-Adam Algorithm
- Convergence Analysis
- Experimental Results
- Conclusions

Conclusions

1) We provide a novel insight to understand adaptive gradient methods by using dynamic mirror descent algorithm;
2) We propose a novel adaptive gradient framework based on dynamic mirror descent algorithm;
3) We provide a convergence analysis framework for our Super-Adam algorithm for nonconvex optimization.

Thanks!

Q&A