ADAPLUS: INTEGRATING NESTEROV MOMENTUM AND PRECISE STEPSIZE ADJUSTMENT ON ADAMW BASIS

Lei Guan

Department of Mathematics, National University of Defense Technology

ABSTRACT

This paper proposes an efficient optimizer called AdaPlus which integrates Nesterov momentum and precise stepsize adjustment on AdamW basis. AdaPlus combines the advantages of AdamW, Nadam, and AdaBelief and, in particular, does not introduce any extra hyper-parameters. We perform extensive experimental evaluations on three machine learning tasks to validate the effectiveness of AdaPlus. The experiment results validate that AdaPlus (i) among all the evaluated adaptive methods, performs most comparable with (even slightly better than) SGD with momentum on image classification tasks and (ii) outperforms other state-of-the-art optimizers on language modeling tasks and illustrates pretty high stability when training GANs. The experiment code of AdaPlus will be accessible at: https://github.com/guanleics/AdaPlus.

Index Terms— deep learning, adaptive method, Nesterov momentum, generalization, stability

1. INTRODUCTION

First-order gradient methods have been broadly used in the training of deep neural networks. The popular first-order gradient methods, in general, can be categorized as accelerated schemes (e.g. stochastic gradient descent with momentum (SGDM) [1]) and adaptive methods (e.g. Adam [2] and AdamW [3]). Adaptive methods generally compute an individual stepsize (a.k.a. learning rate) for each parameter and play a significantly important role in the training of modern deep neural networks. Especially, Adam [2] can attain rapid training speed and has been acting as the default choice for deep learning training.

Much progress on adaptive methods is built upon Adam. For instance, considering the fact that Adam does not generalize as well as SGDM when handling image classification tasks, Loshchilov *et al.* [3] propose the AdamW optimizer which introduces decoupled weight decay into Adam and achieves competitive performance as SGD with momentum when tackling image classification tasks. Based on the observation that Nesterov's accelerated gradient (NAG) [4] is empirically superior to the regular momentum, Timothy Dozat [5] incorporates Nesterov momentum into Adam and proposes the Nadam optimizer. To achieve fast convergence, comparable accuracy to SGD, and provide high stability in the training of a GAN, Zhuang *et al.* [6] propose the AdaBelief optimizer. AdaBelief actually views the exponential moving average (EMA) of the noisy gradient as the prediction of the gradient in the next time step and adapts the stepsize according to the "belief" in the current gradient direction. The advantage of AdaBelief over Adam mainly lies in the " large gradient, small curvature" case where Adabelief, unlike Adam, increases the stepsize as the ideal optimizer does.

It's obvious that AdamW, Nadam, and AdaBlief all build on the basis of Adam but enjoy different advantages in terms of boosting adaptive methods. To combine the benefits of these three adaptive methods, we propose a new optimizer AdaPlus which, on the AdamW basis, simultaneously integrates Nesterov momentum as in Nadam and precise stepsize adjustment as in AdaBelief. To validate the effectiveness of AdaPlus, we experiment with three typical machine learning tasks, including image classification with CNNs on CIFAR10, language modeling with LSTM on Penn TreeBank, and generative adversarial networks (GAN) on CIFAR10. We compare AdaPlus with seven state-of-the-art optimzers including SGDM [1], Adam [2], Nadam [5], RAdam [7], AdamW [3], AdaBelief [6], and AdamW-Win [8]. The experiment results demonstrate that AdaPlus outperforms the other optimizers in simultaneously achieving the goal of (i) fast convergence, (ii) good generalization ability, and (iii) high stability in the training of GANs. For example, on the image classification task, AdaPlus yields an average test accuracy improvement of 1.97% (up to 2.36%), 1.85% (up to 2.0%), and 0.52% (up to 0.89%) over AdamW, Nadam, and AdaBelief, respectively. Furthermore, on the GAN training, AdaPlus always attains a low FID score, illustrating pretty good stability.

The contributions of this paper can be summarized as follows:

(1) We propose a new adaptive optimizer named AdaPlus, which builds based on the AdamW optimizer and further incorporates Nesterov momentum as in Nadam and precise stepsize adjustment as in AdaBelief. AdaPlus is able to combine the advantages of AdamW, Nadam, and AdaBelief. To the best of our knowledge, this is the first adaptive method that simultaneously combines the advantages of decoupled weight decay, Nesterov mo-

mentum, and precise stepsize adjustment.

(2) We conducted extensive experimental evaluations on three different machine-learning tasks to validate the effectiveness of AdaPlus. AdaPlus, among all evaluated optimizers, is the best adaptive method that performs most comparable with SGDM and performs the best in simultaneously achieving the goal of fast convergence, good generalization ability, and high stability.

2. METHODS

Notations In this paper, we let $f(\boldsymbol{\theta}) \in \mathbb{R}^d$ be the loss function to minimize where $\boldsymbol{\theta}$ ($\boldsymbol{\theta} \in \mathbb{R}$) is the parameter to learn. We let $\mathbf{g}_t$ denote the gradient at step t and $\mathbf{m}_t$ refer to the EMA of $\mathbf{g}_t$. The learning rate is represented by a, the weight decay is denoted by u, and ϵ is the smoothing term. Moreover, $\mathbf{v}_t$ and $\mathbf{s}_t$ respectively denote the EMA of $\mathbf{g}_t^2$ and $(\mathbf{g}_t - \mathbf{m}_t)^2$. β_1 and β_2 are the smooting parameters which are typically set to $\beta_1 = 0.9$ and $\beta_2 = 0.999$.

2.1. Modifying AdamW's Momentum

Inspired by [5], we first rewrite NAG as

$$\begin{aligned} &\mathbf{g}_t \leftarrow \nabla_{\boldsymbol{\theta}-1} f(\boldsymbol{\theta}_{t-1}), \\ &\mathbf{m}_t \leftarrow u\mathbf{m}_{t-1} + a\mathbf{g}_t, \\ &\boldsymbol{\theta}_t \leftarrow \boldsymbol{\theta}_{t-1} - (u\mathbf{m}_t + a\mathbf{g}_t). \end{aligned} \tag{1}$$

Equation (1) reveals that NAG updates the parameter with $u\mathbf{m}_t$ rather than $u\mathbf{m}_{t-1}$ used in the classical momentum.

To incorporate Nesterov momentum into AdamW, we replace the classical momentum $\mathbf{m}_t$ in AdamW with Nesterov momentum $\beta_1\mathbf{m}_{t-1} + (1-\beta_1)\mathbf{g}_t$. Then we rewrite AdamW's update step in terms of $\mathbf{m}_{t-1}$ and $\mathbf{g}_t$, which is

$$\boldsymbol{\theta}_t \leftarrow \boldsymbol{\theta}_{t-1} - \frac{a}{\sqrt{\hat{\mathbf{v}}_t} + \epsilon}\Big(\frac{\beta_1\mathbf{m}_{t-1}}{1-\beta_1^t} + \frac{(1-\beta_1)\mathbf{g}_t}{1-\beta_1^t}\Big). \tag{2}$$

After substituting the next momentum step for the current one, we have

$$\boldsymbol{\theta}_t \leftarrow \boldsymbol{\theta}_{t-1} - \frac{a}{\sqrt{\hat{\mathbf{v}}_t} + \epsilon}\Big(\frac{\beta_1\mathbf{m}_t}{1-\beta_1^t} + \frac{(1-\beta_1)\mathbf{g}_t}{1-\beta_1^t}\Big). \tag{3}$$

That can be equivalently rewritten as

$$\begin{aligned} &\bar{\mathbf{m}}_t \leftarrow \beta_1\mathbf{m}_t + (1-\beta_1)\mathbf{g}_t, \\ &\hat{\mathbf{m}}_t \leftarrow \frac{\bar{\mathbf{m}}_t}{1-\beta_1^t}, \\ &\boldsymbol{\theta}_t \leftarrow \boldsymbol{\theta}_{t-1} - \frac{a\hat{\mathbf{m}}_t}{\sqrt{\hat{\mathbf{v}}_t} + \epsilon}. \end{aligned} \tag{4}$$

2.2. Precise Stepsize Adjustment

On the basis of Section 2.1, we further integrate the stepsize adjusting mechanism proposed in [6] and finally propose a new optimizer named AdaPlus. Algorithm 1 summarizes the details of AdaPlus. It's worth noting that no extra hyper-parameters are introduced in AdaPlus in comparison with AdamW and AdaBelief. As shown in Line 7 of Algorithm 1, AdaPlus regards $\mathbf{m}_t$ as the forecast for $\mathbf{g}_t$, escalates the stepsize when $\mathbf{g}_t$ approaches $\mathbf{m}_t$ and decreases the stepsize when $\mathbf{g}_t$ deviates from the prediction $\mathbf{m}_t$.

Algorithm 1 The AdaPlus Optimizer

Require: initial learning rate $a = 0.001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$, weight decay factor $\lambda \in \mathbb{R}$

1: **Initialize** time step $t \leftarrow 0$, $\boldsymbol{\theta}_0$, $\mathbf{m}_0 \leftarrow 0$, $\mathbf{v}_0 \leftarrow 0$, $t \leftarrow 0$.
2: **while** θ_t not converged **do**
3: $t \leftarrow t + 1$
4: $\mathbf{g}_t \leftarrow \nabla_\theta f_t(\boldsymbol{\theta}_{t-1})$
5: $\boldsymbol{\theta}_t \leftarrow \boldsymbol{\theta}_{t-1} - \gamma\lambda\boldsymbol{\theta}_{t-1}$
6: $\mathbf{m}_t \leftarrow \beta_1\mathbf{m}_{t-1} + (1-\beta_1)\mathbf{g}_t$
7: $\mathbf{s}_t \leftarrow \beta_2\mathbf{s}_{t-1} + (1-\beta_2)(\mathbf{g}_t - \mathbf{m}_t)^2 + \epsilon$
8: $\bar{\mathbf{m}}_t \leftarrow \beta_1\mathbf{m}_t + (1-\beta_1)\mathbf{g}_t$
9: $\hat{\mathbf{m}}_t \leftarrow \frac{\bar{\mathbf{m}}_t}{1-\beta_1^t}$, $\hat{\mathbf{s}}_t \leftarrow \frac{\mathbf{s}_t}{1-\beta_2^t}$
10: $\boldsymbol{\theta}_t \leftarrow \boldsymbol{\theta}_{t-1} - \frac{a\hat{\mathbf{m}}_t}{\sqrt{\hat{\mathbf{s}}_t}+\epsilon}$
11: **end while**

Comparison with AdamW, Nadam, and AdaBelief. We mainly consider the "large gradient, small curvature" case in which AdaBelief [6], with precise stepsize adjustment, performs differently from other adaptive methods (e.g. Adam). The details are shown in Figure 1,

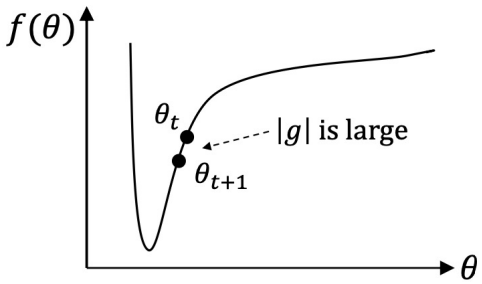


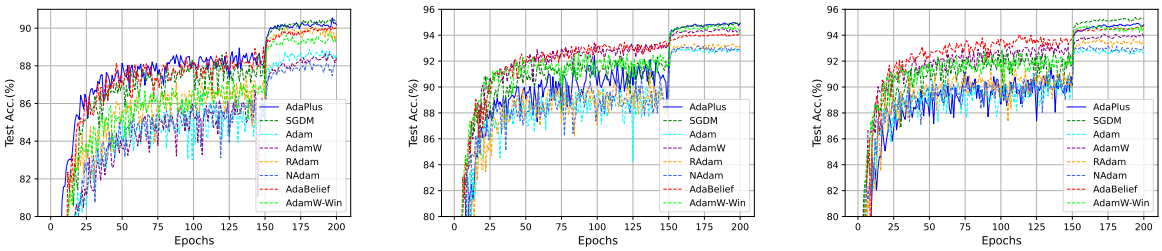
Fig. 1: Illustration of "large gradient, small curvature" case where current stepsize is small and $|g(\boldsymbol{\theta}_t) - g(\boldsymbol{\theta}_{t+1})|$ is small. An ideal optimizer should increase the stepsize.

We note that the update formulas for AdamW, Nadam, AdaBelief, and AdaPlus are:

$$\begin{aligned} \Delta\boldsymbol{\theta}_t^{\text{AdamW, Nadam}} &= -\frac{a\hat{\mathbf{m}}_t}{\sqrt{\hat{\mathbf{v}}_t} + \epsilon}, \\ \Delta\boldsymbol{\theta}_t^{\text{AdaBelief, AdaPlus}} &= -\frac{a\hat{\mathbf{m}}_t}{\sqrt{\hat{\mathbf{s}}_t} + \epsilon} \end{aligned} \tag{5}$$

Table 1: Maximum test accuracy on CIFAR-10. **Higher** is better.

Models	AdaPlus	SGDM	Adam	Nadam	AdamW	RAdam	AdaBelief	AdamW-Win
VGG-11	**90.55**%	90.48%	88.89%	88.19%	88.64%	90.05%	90.07%	89.72%
ResNet-34	**94.99**%	94.96%	92.99%	93.19%	94.50%	93.33%	94.10%	94.72%
DenseNet-121	94.91%	**95.37**%	93.02%	93.17%	94.11%	93.70%	94.71%	94.75%



(a) VGG-11 on CIFAR-10 (b) ResNet-34 on CIFAR-10 (c) DenseNet-121 on CIFAR-10

Fig. 2: Validation accuracy vs. epochs of training VGG-11, ResNet-34, and DenseNet-121 on CIFAR-10.

Equation (5) reveals that the update directions in AdamW and Nadam are $\mathbf{m}_t/(\sqrt{\mathbf{v}_t}+\epsilon)$, where $\mathbf{v}_t$ is the EMA of $\mathbf{g}_t^2$; the update direction in AdaPlus is $\mathbf{m}_t/(\sqrt{\mathbf{s}_t}+\epsilon)$, where $\mathbf{s}_t$ is the EMA of $(\mathbf{g}_t-\mathbf{m}_t)^2$. For the "large gradient, small curvature" case, $|\mathbf{g}_t|$ and $\mathbf{v}_t$ are large, but $|\mathbf{g}_t-\mathbf{g}_{t-1}|$ and $\mathbf{s}_t$ are small. In this case, an ideal optimizer should increase its stepsize. It's clear that AdamW takes a smaller stepsize as $\mathbf{v}_t$ is large. In contrast, as done in an ideal optimizer, AdaPlus and AdaBelief tend to increase its stepsize as $\mathbf{s}_t$ is small. This demonstrates that AdaPlus can take precise stepsize as AdaBelief does.

3. EXPERIMENTS

We perform extensive comparisons with seven state-of-the-art optimizers: SGDM [1], Adam [2], Nadam [5], AdamW [3], RAdam [7], AdaBelief [6], and AdamW-Win [8]. The experimental evaluations include three machine learning tasks, (a) image classification on CIFAR-10 with VGG [9], ResNet [10], and DenseNet [11], (b) language modeling on Penn TreeBank with LSTM [12] models, and (c) Wasserstein-GAN (WGAN) [13] and the improved version with gradient penalty (WGAN-GP) [14] on CIFAR-10 dataset.

We implement AdaPlus in PyTorch on the AdamW basis. The experimental evaluations follow that reported in [6]. On the image classification task, we train all CNN models for 200 epochs with a mini-batch size of 128 and decay the learning rate by 0.1 at the 150th epoch. For the language modeling task, we train LSTMs with 1, 2, and 3 layers on Penn TreeBank dataset where in each experiment, the LSTM models are trained for 200 epochs with a batch size of 20, and the learning rate is decayed by 0.1 at the 100th and 145th epoch.

We note that SGDM, Adam, RAdam, and AdaBelief use the same hyper-parameter tunning strategy as reported [6] which we do not report in detail due to space limit. Nadam and AdamW-Win set their default parameter values in the literature. On the image classification task, we set $\beta_1 = 0.9$ and $\beta_2 = 0.999$. We set weight decay as $1e-2$ and set ϵ as $1e-8$. We initialize the learning rate with 0.001 for VGG-16 and 0.01 for ResNet-34 and DenseNet-121. On the language modeling task, the hyper-parameters for AdaPlus are $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 1e-16$. We initialize the learning rate with $1e-3$ and set the weight decay to $1e-2$. For the training of GANs, we set $a = 2e-4$, $\beta_1 = 0.7$ and $\beta_2 = 0.999$, $\epsilon = 1e-12$, and $\lambda = 1e-2$.

3.1. Experiments for Image Classification

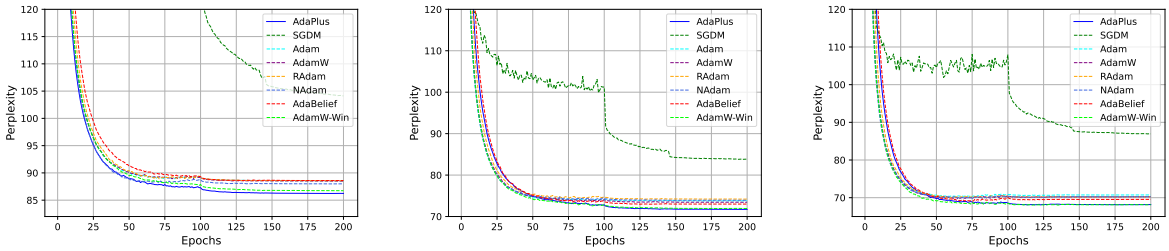
Table 1 summarizes the experiment on the CIFAR-10 dataset. Figure 2 depicts the learning curves of test accuracy vs. epochs for training CNN models of each evaluated optimizer. When training VGG-11 and ResNet-34, AdaPlus always attains higher test accuracy than the other optimizers. In addition, when training DenseNet-121, AdaPlus performs the best among all adaptive methods. In particular, AdaPlus achieves an average of 1.85% (up to 2.0%), 1.97% (up to 2.36%), 1.07% (up to 1.91%), 1.12% (up to 1.66%), 0.52% (up to 0.89%), and 0.42% (up to 0.83%) accuracy improvement over Adam, Nadam, AdamW, RAdam, AdaBelief, and AdamW-Win, respectively.

3.2. Experiments for Language Modeling

Figure 3 depicts the learning curves about perplexity vs. epochs. Table 2 presents the obtained minimum perplexity

Table 2: Minimum perplexity on Penn TreeBank. **Lower** is better.

LSTM	AdaPlus	SGDM	Adam	Nadam	AdamW	RAdam	AdaBelief	AdamW-Win
1 layer	**86.22**	104.13	88.54	87.98	88.49	88.56	88.59	86.73
2 layers	**71.72**	83.80	73.72	73.91	73.43	74.20	72.97	71.93
3 layers	68.08	86.93	70.24	69.82	69.67	70.01	69.10	**68.03**



(a) 1-layer LSTM (b) 2-layer LSTM (c) 3-layer LSTM

Fig. 3: Perplexity vs. epochs of training LSTM on Penn TreeBank.

Table 3: FID (lower is better) of WGAN and WGAN-GP on CIFAR-10.

Model	AdaPlus	SGDM	Adam	Nadam	AdamW	RAdam	AdaBelief	AdamW-Win
WGAN	82.96	299.88	94.15	95.17	93.72	108.09	86.92	**60.10**
WGAN-GP	**63.70**	257.67	76.60	76.54	68.85	94.29	66.63	64.40

(lower is better). The experimental results shown in Table 2 again validate the generalization ability of AdaPlus. When training the 1-layer and 2-layer LSTM models, AdaPlus consistently attains the lowest perplexity among all evaluated optimizers. For training 3-layer LSTM, AdaPlus ranks second but demonstrates pretty comparable perplexity with AdamW-Win.

3.3. Experiments for GANs on CIFAR-10

In this section, we experiment with the Wasserstein-GAN (WGAN) [13] and WGAN-GP [14]. As reported in [6], using each optimizer, we train the model for 100 epochs, generating 64,000 fake images from noise. We compute the Frechet Inception Distance (FID) score between the fake images and the real dataset to assess the generative models. Table 3 reports the final FID score (lower is better). AdaPlus gets the second-lowest FID score when training WGAN and achieves the lowest FID score when training WGAN-GP. In particular, AdaPlus also outperforms AdaBelief (82.96 vs. 86.92 for WGAM and 63.70 vs. 66.63 for WGAN-GP), which demonstrates that aside from precise stepsize adjustment, simultaneously integrating Nesterov momentum and decoupled weight decay helps boost the stability when training GANs.

4. RELATED WORK

Unlike SGDM [1], adaptive methods dynamically scale the gradient according to the EMA of the past gradients. Representative adaptive methods include AdaGrad [15], RMSprop [16], and Adam [2], which enjoy fast speed in the early training period yet exhibit poorer generalization ability than SGD. Apart from Nadam [5], AdamW [3], and AdaBelief [6], other variants of Adam also have been proposed (e.g., Yogi [17], RAdam [7], AMSGrad [18], AdaBound [19], AdaMomentum [20], Adan [21], and Lion [22]). These adaptive methods target to achieve the same goal —accelerating the training and improving the generalization at the same time. Very recently, XGrad [23] was proposed which incorporates weight prediction [24] into the DNN training in an effort to boost the convergence and generalization of gradient-based optimizers.

5. CONCLUSIONS

This paper proposes a novel and efficient adaptive method AdaPlus which combines the benefits of AdamW, Nadam, and AdaBelief and does not introduce any extra parameters. The experiment evaluations demonstrate that AdaPlus outperforms the other seven state-of-the-art optimizers in terms of convergence, generalization ability, and stability.

6. REFERENCES

[1] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton, "On the importance of initialization and momentum in deep learning," in *International conference on machine learning*. PMLR, 2013, pp. 1139–1147.

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

[3] Ilya Loshchilov and Frank Hutter, "Decoupled weight decay regularization," *arXiv preprint arXiv:1711.05101*, 2017.

[4] Y Nesterov, "A method of solving a convex programming problem with convergence rate mathcal {O}(1/k {2})," in *Sov. Math. Dokl*, vol. 27.

[5] Timothy Dozat, "Incorporating nesterov momentum into adam," 2016.

[6] Juntang Zhuang, Tommy Tang, Yifan Ding, Sekhar C Tatikonda, Nicha Dvornek, Xenophon Papademetris, and James Duncan, "Adabelief optimizer: Adapting stepsizes by the belief in observed gradients," *Advances in neural information processing systems*, vol. 33, pp. 18795–18806, 2020.

[7] Liyuan Liu, Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Jiawei Han, "On the variance of the adaptive learning rate and beyond," *arXiv preprint arXiv:1908.03265*, 2019.

[8] Pan Zhou, Xingyu Xie, and YAN Shuicheng, "Win: Weight-decay-integrated nesterov acceleration for adaptive gradient algorithms," in *The Eleventh International Conference on Learning Representations*, 2022.

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

[10] 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*, 2016, pp. 770–778.

[11] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 4700–4708.

[12] Xiaolei Ma, Zhimin Tao, Yinhai Wang, Haiyang Yu, and Yunpeng Wang, "Long short-term memory neural network for traffic speed prediction using remote microwave sensor data," *Transportation Research Part C: Emerging Technologies*, vol. 54, pp. 187–197, 2015.

[13] Martin Arjovsky, Soumith Chintala, and Léon Bottou, "Wasserstein generative adversarial networks," in *International conference on machine learning*. PMLR, 2017, pp. 214–223.

[14] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen, "Improved techniques for training gans," *Advances in neural information processing systems*, vol. 29, 2016.

[15] John Duchi, Elad Hazan, and Yoram Singer, "Adaptive subgradient methods for online learning and stochastic optimization.," *Journal of machine learning research*, vol. 12, no. 7, 2011.

[16] Tijmen Tieleman and Geoffrey Hinton, "Lecture 6.5-rmsprop, coursera: Neural networks for machine learning," *University of Toronto, Technical Report*, vol. 6, 2012.

[17] Manzil Zaheer, Sashank Reddi, Devendra Sachan, Satyen Kale, and Sanjiv Kumar, "Adaptive methods for nonconvex optimization," in *Advances in Neural Information Processing Systems*, 2018, vol. 31, pp. 9815–9825.

[18] Sashank J Reddi, Satyen Kale, and Sanjiv Kumar, "On the convergence of adam and beyond," *arXiv preprint arXiv:1904.09237*, 2019.

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

[20] Yizhou Wang, Yue Kang, Can Qin, Huan Wang, Yi Xu, Yulun Zhang, and Yun Fu, "Rethinking adam: A twofold exponential moving average approach," *arXiv preprint arXiv:2106.11514*, 2021.

[21] Xingyu Xie, Pan Zhou, Huan Li, Zhouchen Lin, and Shuicheng Yan, "Adan: Adaptive nesterov momentum algorithm for faster optimizing deep models," *arXiv preprint arXiv:2208.06677*, 2022.

[22] Xiangning Chen, Chen Liang, Da Huang, Esteban Real, Kaiyuan Wang, Yao Liu, Hieu Pham, Xuanyi Dong, Thang Luong, Cho-Jui Hsieh, et al., "Symbolic discovery of optimization algorithms," *arXiv preprint arXiv:2302.06675*, 2023.

[23] Lei Guan, Dongsheng Li, Jian Meng, and Yanqi Shi, "Xgrad: Boosting gradient-based optimizers with weight prediction," *arXiv preprint arXiv:2305.18240*, 2023.

[24] Lei Guan, Wotao Yin, Dongsheng Li, and Xicheng Lu, "Xpipe: Efficient pipeline model parallelism for multi-gpu dnn training," *arXiv preprint arXiv:1911.04610*, 2019.