Appendix: On the Overlooked Pitfalls of Weight Decay and How to Mitigate Them

Zeke Xie[1,2], Zhiqiang Xu[3], Jingzhao Zhang[4], Issei Sato[1], and Masashi Sugiyama[2,1]

[1]The University of Tokyo
[2]RIKEN Center for AIP
[3]MBZUAI
[4]Tsinghua University
Correspondence: *zekexie16@gmail.com*, *Zhiqiang.Xu@mbzuai.ac.ae*

A Proofs

A.1 Proof of Theorem 1

Proof. We first write the regularized loss function corresponding to SGD with vanilla weight decay as

$$f_t(\theta) = L(\theta) + \frac{\lambda'}{2\eta_t}\|\theta\|^2 \tag{1}$$

at t-th step.

If the corresponding L_2 regularization $\frac{\lambda'}{2\eta_t}$ is unstable during training, the regularized loss function $f_t(\theta)$ will also be a time-dependent function and has no non-zero stable stationary points.

Suppose we have a non-zero solution $\theta^\star$ which is a stationary point of $f(\theta, t)$ at t-th step and SGD finds $\theta_t = \theta^\star$ at t-th step.

Even if the gradient of $f_t(\theta)$ at t-step is zero, we have the gradient at $(t+1)$-th step as

$$g_{t+1} = \nabla f_{t+1}(\theta^\star) = \lambda'(\eta_t^{-1} - \eta_{t+1}^{-1})\theta^\star. \tag{2}$$

It means that

$$\|g_{t+1}\|^2 = \lambda'^2(\eta_t^{-1} - \eta_{t+1}^{-1})^2\|\theta^\star\|^2 \geq \frac{\lambda'^2\delta^2\|\theta^\star\|^2}{\eta_t\eta_{t+1}} \tag{3}$$

To achieve convergence, we must have $\|g_{t+1}\|^2 = 0$.

It requires $(\eta_t^{-1} - \eta_{t+1}^{-1})^2 = 0$ or $\|\theta^\star\|^2 = 0$.

Theorem 2.2 of Shapiro and Wardi [9] told us that the learning rate should be small enough for convergence. Obviously, we have $\eta < \infty$ in practice.

As $\eta_t = \eta_{t+1}$ does not hold, SGD cannot converging to any non-zero stationary point.

The proof is now complete. □

A.2 Proof of Theorem 2

Before formally completing the proof, we first introduce a useful Lemma 1, which is a specialized case of Theorem 1 in Yan et al. [10] with $\beta = s = 0$.

37th Conference on Neural Information Processing Systems (NeurIPS 2023).

Lemma 1 (Convergence of SGD)**.** *Assume that $L(\theta)$ is an $\mathcal{L}$-smooth function*[1]*, L is lower bounded as $L(\theta) \geq L^\star$, $\mathbb{E}[\nabla L(\theta, X) - \nabla L(\theta)] = 0$, $\mathbb{E}[\|\nabla L(\theta, X) - \nabla L(\theta)\|^2] \leq \delta^2$, $\|\nabla L(\theta)\| \leq G$ for any θ. Let SGD optimize L for $t+1$ iterations. If $\eta \leq \frac{C}{\sqrt{t+1}}$, we have*

$$\min_{k=0,\ldots,t} \mathbb{E}[\|\nabla L(\theta_k)\|^2] \leq \frac{C_0}{\sqrt{t+1}}, \tag{4}$$

where $C_0 = \left[\frac{L(\theta_0)-L^\star}{C} + C\mathcal{L}(G^2+\sigma^2)\right]$.

Proof. Given the conditions of $L(\theta)$ in Lemma 1, we may obtain the resulted conditions of $f(\theta) = L(\theta) + \frac{\lambda}{2}\|\theta\|^2$.

As $L(\theta)$ is an $\mathcal{L}$-smooth function, we have

$$\|\nabla f(\theta_a) - \nabla f(\theta_b)\| = \|\nabla L(\theta_a) - \nabla L(\theta_b) + \lambda(\theta_a - \theta_b)\| \leq (\mathcal{L}+\lambda)\|\theta_a - \theta_b\| \tag{5}$$

holds for any θ_a and θ_b. It shows that $f(\theta)$ is an $(\mathcal{L}+\lambda)$-smooth function.

As L is lower bounded as $L(\theta) \geq L^\star$, we have

$$f^\star \geq L^\star. \tag{6}$$

As $\mathbb{E}[\nabla L(\theta, X) - \nabla L(\theta)] = 0$, we have

$$\mathbb{E}[\nabla f(\theta, X) - \nabla f(\theta)] = \mathbb{E}[\nabla L(\theta, X) - \nabla L(\theta)] = 0. \tag{7}$$

As $\mathbb{E}[\|\nabla L(\theta, X) - \nabla L(\theta)\|^2] \leq \delta^2$, we have

$$\mathbb{E}[\|\nabla f(\theta, X) - \nabla f(\theta)\|^2] = \mathbb{E}[\|\nabla L(\theta, X) - \nabla L(\theta)\|^2] \leq \delta^2. \tag{8}$$

As $\|\nabla L(\theta)\| \leq G$, we have

$$\|\nabla f(\theta)\| = \|\nabla L(\theta) + \lambda\theta\| \leq G + \lambda\|\theta\|_{\max}, \tag{9}$$

where $\|\theta\|_{\max}$ is the maximum L_2 norm of any θ.

Introducing the derived conditions Eq. (12) - (16) for f into Lemma 1, we may treat f as the objective optimized by SGD. Then we have

$$\min_{k=0,\ldots,t} \mathbb{E}[\|\nabla f(\theta_k)\|^2] \leq \frac{1}{\sqrt{t+1}}\left[\frac{f(\theta_0) - f^\star}{C} + C(\mathcal{L}+\lambda)((G+\lambda\|\theta\|_{\max})^2 + \sigma^2)\right] \tag{10}$$

$$\leq \frac{1}{\sqrt{t+1}}\left[\frac{L(\theta_0) + \frac{\lambda}{2}\|\theta_0\|^2 - L^\star}{C} + C(\mathcal{L}+\lambda)((G+\lambda\|\theta\|_{\max})^2 + \sigma^2)\right] \tag{11}$$

Obviously, the gradient norm upper bound in convergence analysis monotonically increases as the weight decay strength λ.

The proof is complete.

□

B Experimental Details

Computational environment. The experiments are conducted on a computing cluster with GPUs of NVIDIA® Tesla™ P100 16GB and CPUs of Intel® Xeon® CPU E5-2640 v3 @ 2.60GHz.

[1] It means that $\|\nabla L(\theta_a) - \nabla L(\theta_b)\| \leq \mathcal{L}\|\theta_a - \theta_b\|$ holds for any θ_a and θ_b.

B.1 Image Classification on CIFAR-10 and CIFAR-100

Data Preprocessing For CIFAR-10 and CIFAR-100: We perform the common per-pixel zero-mean unit-variance normalization, horizontal random flip, and 32×32 random crops after padding with 4 pixels on each side.

Hyperparameter Settings: We select the optimal learning rate for each experiment from $\{0.0001, 0.001, 0.01, 0.1, 1, 10\}$ for non-adaptive gradient methods. We use the default learning rate for adaptive gradient methods in the experiments of Table 1, while we also compared Adam, AdamW, AdamS under various learning rates and batch sizes in other experiments. In the experiments on CIFAF-10 and CIFAR-100: $\eta = 0.1$ for SGD and SGDS; $\eta = 0.001$ for Adam, AdamW, AdamS, AMSGrad, Yogi, AdaBound, and RAdam; $\eta = 0.01$ for Padam. For the learning rate schedule, the learning rate is divided by 10 at the epoch of $\{80, 160\}$ for CIFAR-10 and $\{100, 150\}$ for CIFAR-100, respectively. The batch size is set to 128 for both CIFAR-10 and CIFAR-100.

The strength of L_2 regularization and SWD is default to 0.0005 as the baseline. Considering the linear scaling rule, we choose $\lambda_W = \frac{\lambda_{L_2}}{\eta}$. Thus, the weight decay of AdamW uses $\lambda_W = 0.5$ for CIFAR-10 and CIFAR-100. The basic principle of choosing weight decay strength is to let all optimizers have similar convergence speed.

We set the momentum hyperparameter $\beta_1 = 0.9$ for SGD and SGDS. As for other optimizer hyperparameters, we apply the default hyperparameter settings directly.

We repeated each experiment for three times in the presence of the error bars.

We leave the empirical results with the weight decay setting $\lambda = 0.0001$ in Appendix C.

B.2 Image classification on ImageNet

Data Preprocessing For ImageNet: For ImageNet, we perform the per-pixel zero-mean unit-variance normalization, horizontal random flip, and the resized random crops where the random size (of 0.08 to 1.0) of the original size and a random aspect ratio (of $\frac{3}{4}$ to $\frac{4}{3}$) of the original aspect ratio is made.

Hyperparameter Settings for ImageNet: We select the optimal learning rate for each experiment from $\{0.0001, 0.001, 0.01, 0.1, 1, 10\}$ for all tested optimizers. For the learning rate schedule, the learning rate is divided by 10 at the epoch of $\{30, 60\}$. We train each model for 90 epochs. The batch size is set to 256. The weight decay hyperparameter of AdamS, AdamW, Adam are chosen from $\{5 \times 10^{-6}, 5 \times 10^{-5}, 5 \times 10^{-4}, 5 \times 10^{-3}, 5 \times 10^{-2}\}$. As for other optimizer hyperparameters, we still apply the default hyperparameter settings directly.

B.3 Language Modeling

We use a classical language model, Long Short-Term Memory (LSTM) [2] with 2 layers, 512 embedding dimensions, and 512 hidden dimensions, which has 14 million model parameters and is similar to the "medium LSTM" in Zaremba et al. [12]. Note that our baseline performance is better than the reported baseline performance in Zaremba et al. [12]. The benchmark task is the word-level Penn TreeBank [7]. We empirically compared AdamS, AdamW, and Adam under the common and same conditions.

Hyperparameter Settings. Batch Size: $B = 20$. BPTT Size: $bptt = 35$. Learning Rate: $\eta = 0.001$. We run the experiments under various weight decay selected from $\{10^{-4}, 5 \times 10^{-5}, 10^{-5}, 5 \times 10^{-6}, 10^{-6}, 5 \times 10^{-7}, 10^{-7}\}$.The dropout probability is set to 0.5. We clipped gradient norm to 1.

C Supplementary Figures and Results of Adaptive Gradient Methods

Popular Adam variants often generalize worse than SGD. A few Adam variants tried to fix the hidden problems in adaptive gradient methods, including AdamW Loshchilov and Hutter [5], AMSGrad [8] and Yogi [11]. A recent line of research, such as AdaBound [6], Padam [1], and RAdam [3], believes controlling the adaptivity of learning rates may improve generalization. This line of research usually introduces extra hyperparameters to control the adaptivity, which requires more efforts in tuning hyperparameters. However, we and Zhang et al. [13] found that this argument

Table 1: Test performance comparison of optimizers with $\lambda_{L_2} = \lambda_S = 0.0001$ and $\lambda_W = 0.1$, which is a common weight decay setting in related papers. AdamS still show better test performance than popular adaptive gradient methods and SGD.

Dataset	Model	SGD	AdamS	Adam	AMSGrad	AdamW	AdaBound	Padam	Yogi	RAdam
CIFAR-10	ResNet18	5.58	**4.69**	6.08	5.72	5.33	6.87	5.83	5.43	5.81
	VGG16	6.92	**6.16**	7.04	6.68	6.45	7.33	6.74	6.69	6.73
CIFAR-100	ResNet34	24.92	**23.50**	25.56	24.74	23.61	25.67	25.39	23.72	25.65
	DenseNet121	**20.98**	21.35	24.39	22.80	22.23	24.23	22.26	22.40	22.40
	GoogLeNet	21.89	**21.60**	24.60	24.05	21.71	25.03	26.69	22.56	22.35

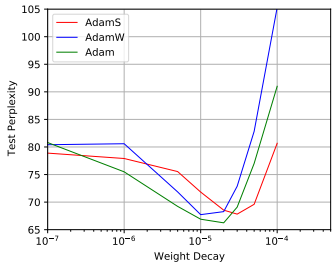


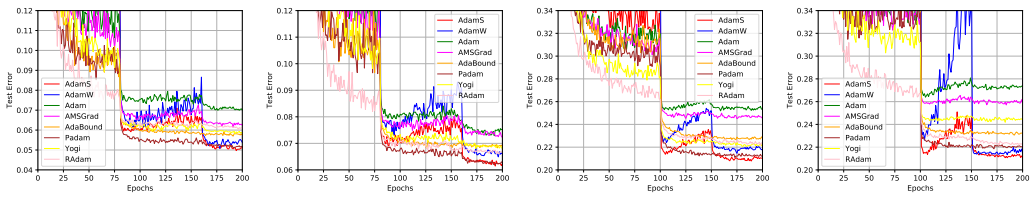
Figure 1: Language modeling under various weight decay. Note that the lower perplexity is better.

is contradicted with our comparative experimental results (see Table 1). In our empirical analysis, most advanced Adam variants may narrow but not completely close the generalization gap between adaptive gradient methods and SGD. SGD with a fair weight decay hyperparameter as the baseline performance usually generalizes better than recent adaptive gradient methods. The main problem may lie in weight decay. SGD with weight decay $\lambda = 0.0001$, a common setting in related papers, is often not a good baseline, as $\lambda = 0.0005$ often shows better generalization on CIFAR-10 and CIFAR-100. We also conduct comparative experiments with $\lambda = 0.0001$. Under the setting $\lambda = 0.0001$, while some existing Adam variants may outperform SGD sometimes due to the lower baseline performance of SGD, AdamS shows superior test performance. For example, for ResNet18 on CIFAR-10, the test error of AdamS is lower than SGD by nearly one point and no other Adam variant may compare with AdamS.

Language Modeling. It is well-known that, different from computer vision tasks, the standard Adam (with L_2 regularization) is the most popular optimizer for language models. Figure 1 in Appendix C demonstrates that the conventional belief is true that the standard L_2 regularization yields better test results than both Decoupled Weight Decay and SWD. The weight decay scheduler suitable for language models is an open problem.

We report the learning curves of all adaptive gradient methods in Figure 2. They shows that vanilla Adam with SWD can outperform other complex variants of Adam.

Figure 3 displays the scatter plot of training losses and test errors during final 40 epochs of training DenseNet121 on CIFAR-100.



(a) ResNet18 (b) VGG16 (c) DenseNet121 (d) GoogLeNet

Figure 2: The learning curves of adaptive gradient methods.

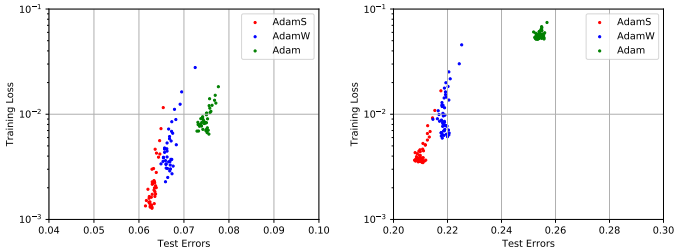


Figure 3: Even if with similar or higher training losses, AdamS still generalizes better than AdamW and Adam. The scatter plot of training losses and test errors during final 50 epochs of training VGG16 on CIFAR-10 and DenseNet121 on CIFAR-100.

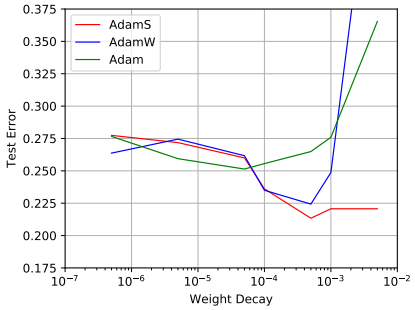


Figure 4: We compare the generalization of Adam, AdamW, and AdamS with various weight decay rates by training ResNet34 on CIFAR-100. The displayed weight decay of AdamW in the figure has been rescaled by the factor $= 0.001$. The optimal test performance of AdamS is significantly better than AdamW and Adam.

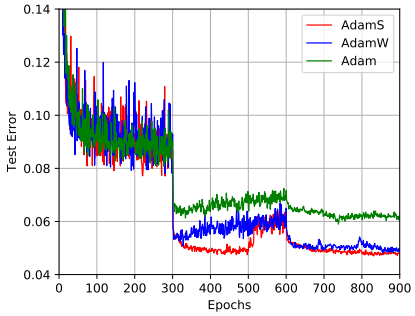


Figure 5: We train ResNet18 on CIFAR-10 for 900 epochs to explore the performance limit of AdamS, AdamW, and Adam. The learning rate is divided by 10 at the epoch of 300 and 600. AdamS achieves the most optimal test error, 4.70%.

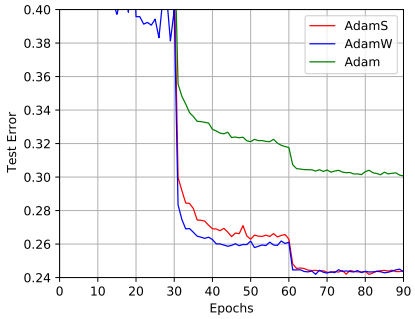


Figure 6: ResNet50 on ImageNet. The lowest Top-1 test errors of AdamS, AdamW, and Adam are 24.19%, 24.29%, and 30.07%, respectively.

Table 2: In the experiment of ResNet18 trained via SGD on CIFAR-10, we verified that the optimal weight decay is approximately inverse to the number of epochs. The predicted optimal weight decay is approximately $0.1 \times \text{Epochs}^{-1}$, because the optimal weight decay is $\lambda = 0.0005$ selected from $\{10^{-2}, 5 \times 10^{-3}, 10^{-3}, 5 \times 10^{-4}, 10^{-4}, 5 \times 10^{-5}, 10^{-5}, 5 \times 10^{-6}, 10^{-6}\}$ with 200 epochs as the base case. The observed optimal weight decay is selected from $\{\text{Epochs}^{-1}, 0.1 \times \text{Epochs}^{-1}, 0.01 \times \text{Epochs}^{-1}\}$. We observed that the optimal test errors are all corresponding to the predicted optimal weight decay $\lambda = 0.1 \times \text{Epochs}^{-1}$. At least in the sense of the order of magnitude, the predicted optimal weight decay is fully consistent with the observed optimal weight decay. Thus, the empirical results supports that the optimal weight decay is approximately inverse to the number of epochs in the common range of the number of epochs.

EPOCHS	$\lambda = \text{Epochs}^{-1}$	$\lambda = 0.1 \times \text{Epochs}^{-1}$	$\lambda = 0.01 \times \text{Epochs}^{-1}$
50	74.06	**7.12**	7.50
100	22.04	**5.56**	6.01
200	11.81	**5.02**	5.61
1000	4.67	**4.43**	6.02
2000	4.59	**4.48**	5.70

Figure 4 displays the test performance of AdamS, AdamW, and Adam under various weight decay hyperparameters of ResNet34 on CIFAR-100.

We train ResNet18 on CIFAR-10 for 900 epochs to explore the performance limit of AdamS, AdamW, and Adam in Figure 5.

Figure 6 in Appendix shows that, AdamS can make marginal improvements over AdamW.

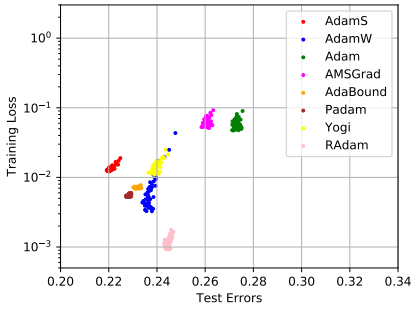


Figure 7: The scatter plot of training losses and test errors during final 40 epochs of training ResNet34 on CIFAR-100. Even with similar or higher training losses, AdamS still generalizes better than other Adam variants. We leave the scatter plot on CIFAR-10 in Appendix C.

D Additional Algorithms

We note that the implementation of AMSGrad in Algorithm 1 is the popular implementation in PyTorch. We use the PyTorch implementation in our paper, as it is widely used in practice.

Algorithm 1: AMSGrad/AMSGradW

$g_t = \nabla L(\theta_{t-1}) + \lambda\theta_{t-1}$;
$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t$;
$v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2$;
$\hat{m}_t = \frac{m_t}{1-\beta_1^t}$;
$v_{max} = max(v_t, v_{max})$;
$\hat{v}_t = \frac{v_{max}}{1-\beta_2^t}$;
$\theta_t = \theta_{t-1} - \frac{\eta}{\sqrt{\hat{v}_t}+\epsilon}\hat{m}_t - \eta\lambda\theta_{t-1}$;

Algorithm 2: AMSGradS

$g_t = \nabla L(\theta_{t-1})$;
$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t$;
$v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2$;
$\hat{m}_t = \frac{m_t}{1-\beta_1^t}$;
$v_{max} = max(v_t, v_{max})$;
$\hat{v}_t = \frac{v_{max}}{1-\beta_2^t}$;
$\bar{v}_t = mean(\hat{v}_t)$;
$\theta_t = \theta_{t-1} - \frac{\eta}{\sqrt{\hat{v}_t}+\epsilon}\hat{m}_t - \frac{\eta}{\sqrt{\bar{v}_t}}\lambda\theta_{t-1}$;

E Supplementary Experiments with Cosine Annealing Schedulers and Warm Restarts

In this section, we conducted comparative experiments on AdamS, AdamW, and Adam in the presence of cosine annealing schedulers and warm restarts proposed by Loshchilov and Hutter [4]. We set the learning rate scheduler with a recommended setting of Loshchilov and Hutter [4]: $T_0 = 14$ and $T_{mul} = 2$. The number of total epochs is 210. Thus, we trained each deep network for four runs of warm restarts, where the four runs have 14, 28, 56, and 112 epochs, respectively. Other hyperparameters and details are displayed in Appendix B.

We conducted comparative experiments on AdamS, AdamW, and Adam in the presence of cosine annealing schedulers and warm restarts proposed by Loshchilov and Hutter [4]. We set the learning rate scheduler with a recommended setting of Loshchilov and Hutter [4] Our experimental results in Figures 11 and 8 suggest that AdamS consistently outperforms AdamW and Adam in the presence of cosine annealing schedulers and warm restarts. It demonstrates that, with various learning rate schedulers, the advantage of SWD may generally hold.

Moreover, we did not empirically observe that cosine annealing schedulers with warm restarts may consistently outperform the common piecewise-constant learning rate schedulers for adaptive gradient methods. We noticed that Loshchilov and Hutter [4] empirically compared four-staged piecewise-constant learning rate schedulers with cosine annealing schedulers with warm restarts, and argue that cosine annealing schedulers with warm restarts are better. There may be two possible causes. First, three-staged piecewise-constant learning rate schedulers, which usually have a longer first stage and decay learning rates by multiplying 0.1, are the recommended settings, while the four-staged piecewise-constant learning rate schedulers in Loshchilov and Hutter [4] are usually not optimal. Second, warm restarts may be helpful, while cosine annealing may be not. The ablation study on piecewise-constant learning rate schedulers with warm restarts is lacked. We argued that how to choose learning rate schedulers may still be an open question, considering the complex choices of schedulers and the complex loss landscapes.

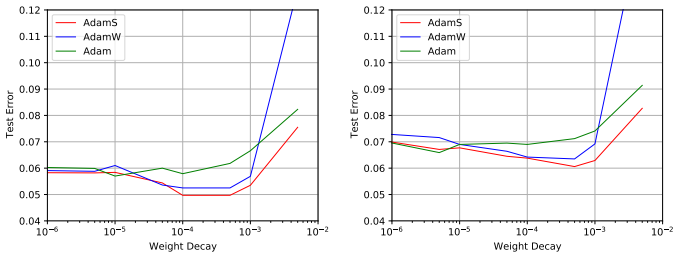
References

[1] Chen, J. and Gu, Q. (2018). Closing the generalization gap of adaptive gradient methods in training deep neural networks. *arXiv preprint arXiv:1806.06763*.

[2] Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8):1735–1780.

[3] Liu, L., Jiang, H., He, P., Chen, W., Liu, X., Gao, J., and Han, J. (2019). On the variance of the adaptive learning rate and beyond. In *International Conference on Learning Representations*.

[4] Loshchilov, I. and Hutter, F. (2016). Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*.



(a) ResNet18 (b) VGG16

Figure 8: The test errors of ResNet18 and VGG16 on CIFAR-10 under various weight decay with cosine annealing and warm restart schedulers. AdamS yields significantly better optimal test performance than AdamW and Adam.

[5] Loshchilov, I. and Hutter, F. (2018). Decoupled weight decay regularization. In *International Conference on Learning Representations*.

[6] Luo, L., Xiong, Y., Liu, Y., and Sun, X. (2019). Adaptive gradient methods with dynamic bound of learning rate. *7th International Conference on Learning Representations, ICLR 2019*.

[7] Marcus, M., Santorini, B., and Marcinkiewicz, M. A. (1993). Building a large annotated corpus of english: The penn treebank.

[8] Reddi, S. J., Kale, S., and Kumar, S. (2019). On the convergence of adam and beyond. *6th International Conference on Learning Representations, ICLR 2018*.

[9] Shapiro, A. and Wardi, Y. (1996). Convergence analysis of gradient descent stochastic algorithms. *Journal of optimization theory and applications*, 91(2):439–454.

[10] Yan, Y., Yang, T., Li, Z., Lin, Q., and Yang, Y. (2018). A unified analysis of stochastic momentum methods for deep learning. In *IJCAI International Joint Conference on Artificial Intelligence*.

[11] Zaheer, M., Reddi, S., Sachan, D., Kale, S., and Kumar, S. (2018). Adaptive methods for nonconvex optimization. In *Advances in neural information processing systems*, pages 9793–9803.

[12] Zaremba, W., Sutskever, I., and Vinyals, O. (2014). Recurrent neural network regularization. *arXiv preprint arXiv:1409.2329*.

[13] Zhang, J., He, T., Sra, S., and Jadbabaie, A. (2019). Why gradient clipping accelerates training: A theoretical justification for adaptivity. In *International Conference on Learning Representations*.