–Supplemental Document–
Exploiting Explainable Metrics for Augmented SGD

Mahdi S. Hosseini[1*] Mathieu Tuli[2*] Konstantinos N. Plataniotis[2]
[1]University of New Brunswick [2]University of Toronto

Code:https://github.com/mahdihosseini/RMSGD

Contents

*Equally major contribution

Appendix-A: Proof of Theorems and Remarks

1. Proofs for Theorem 1

The following two proofs correspond to the proof of Theorem 1 for $p = 2$ and $p = 1$, respectively. Note we have omitted layer index ℓ for convenience. (Theorem 1) Recall that the summation of squared singular values of a matrix is equivalent to the *Frobenius* (norm) i.e. $||\mathbf{W}^t||_F^2 = \sum_{k=1}^{N'_d} \sigma_k^2(\mathbf{W}^t) = \text{Tr}\left\{{\mathbf{W}^t}^T \mathbf{W}^t\right\}$ [3]. Using the definition of stable-rank in Equation (2) from main paper draft, Section 4.1, the stable rank of matrix $\mathbf{W}^{t+1}$ (assumed to be a column matrix $m \leq n$) is expressed by

$$s(\mathbf{W}^{t+1}) = \frac{1}{n\sigma_1^2(\mathbf{W}^{t+1})} \sum_{i=1}^{n'} \sigma_i^2(\mathbf{W}^{t+1}) = \frac{1}{n||\mathbf{W}^{t+1}||_2^2} \text{Tr}\left\{{\mathbf{W}^{t+1}}^T \mathbf{W}^{t+1}\right\}. \tag{1}$$

An upper-bound of first singular value can be calculated by first recalling its equivalence to ℓ_2-norm and then applying the Cauchy–Schwarz inequality

$$\sigma_1^2(\mathbf{W}^{t+1}) = ||\mathbf{W}^{t+1}||_2^2 = ||\mathbf{W}^t - \eta_\ell(t)\overline{\nabla f}_{t+1}||_2^2 \leq ||\mathbf{W}^t||_2^2 + \eta_\ell^2(t)||\overline{\nabla f}_{t+1}||_2^2 + 2\eta_\ell(t)||\mathbf{W}^t||_2||\overline{\nabla f}_{t+1}||_2. \tag{2}$$

Note that $\eta_\ell(t)$ is given by previous epoch update and considered to be positive (we start with an initial learning rate $\eta(0) > 0$). Therefore, the right-hand-side of the inequality in (2) will be positive and holds.

By substituting (2) in (1) and expanding the terms in trace, a lower bound of $\mathbf{W}^{t+1}$ is given by

$$s(\mathbf{W}^{t+1}) \geq \frac{1}{N\gamma}\left[\text{Tr}\left\{{\mathbf{W}^t}^T \mathbf{W}^t\right\} - 2\eta_\ell(t)\text{Tr}\left\{{\mathbf{W}^t}^T \overline{\nabla f}_{t+1}\right\} + \eta_\ell^2(t)\text{Tr}\left\{{\overline{\nabla f}_{t+1}}^T \overline{\nabla f}_{t+1}\right\}\right], \tag{3}$$

where, $\gamma = ||\mathbf{W}^t||_2^2 + \eta_\ell^2(t)||\overline{\nabla f}_{t+1}||_2^2 + 2\eta_\ell(t)||\mathbf{W}^t||_2||\overline{\nabla f}_{t+1}||_2$. The latter inequality can be revised to

$$\begin{aligned}
s(\mathbf{W}^{t+1}) &\geq \tfrac{1}{N\gamma}\left[\left(1 - \tfrac{\gamma}{||\mathbf{W}^t||_2^2} + \tfrac{\gamma}{||\mathbf{W}^t||_2^2}\right)\text{Tr}\left\{{\mathbf{W}^t}^T \mathbf{W}^t\right\}\right. \\
&\qquad \left. -2\eta_\ell(t)\text{Tr}\left\{{\mathbf{W}^t}^T \overline{\nabla f}_{t+1}\right\} + \eta_\ell^2(t)\text{Tr}\left\{{\overline{\nabla f}_{t+1}}^T \overline{\nabla f}_{t+1}\right\}\right] \\
&= \tfrac{1}{N\gamma}\left[\tfrac{\gamma}{||\mathbf{W}^t||_2^2}\text{Tr}\left\{{\mathbf{W}^t}^T \mathbf{W}^t\right\} + \left(1 - \tfrac{\gamma}{||\mathbf{W}^t||_2^2}\right)\text{Tr}\left\{{\mathbf{W}^t}^T \mathbf{W}^t\right\}\right. \\
&\qquad \left. -2\eta_\ell(t)\text{Tr}\left\{{\mathbf{W}^t}^T \overline{\nabla f}_{t+1}\right\} + \eta_\ell^2(t)\text{Tr}\left\{{\overline{\nabla f}_{t+1}}^T \overline{\nabla f}_{t+1}\right\}\right] \\
&= s(\mathbf{W}^t) + \tfrac{1}{N\gamma}\left[\underbrace{\left(1 - \frac{\gamma}{||\mathbf{W}^t||_2^2}\right)\text{Tr}\left\{{\mathbf{W}^t}^T \mathbf{W}^t\right\} - 2\eta_\ell(t)\text{Tr}\left\{{\mathbf{W}^t}^T \overline{\nabla f}_{t+1}\right\} + \eta_\ell^2(t)\text{Tr}\left\{{\overline{\nabla f}_{t+1}}^T \overline{\nabla f}_{t+1}\right\}}_{D}\right].
\end{aligned} \tag{4}$$

Therefore, the bound in (4) is revised to

$$s(\mathbf{W}^{t+1}) - s(\mathbf{W}^t) \geq \frac{1}{N\gamma} D. \tag{5}$$

Since $\gamma \geq 0$, the monotonicity of the Equation (5) is guaranteed if $D \geq 0$. The remaining term D can be expressed as a quadratic function of η

$$\begin{aligned}
D(\eta) = &\left[\text{Tr}\left\{{\overline{\nabla f}_{t+1}}^T \overline{\nabla f}_{t+1}\right\} - \tfrac{||\overline{\nabla f}_{t+1}||_2^2}{||\mathbf{W}^t||_2^2}\text{Tr}\left\{{\mathbf{W}^t}^T \mathbf{W}^t\right\}\right]\eta_\ell^2(t) \\
&- \left[2\text{Tr}\left\{{\mathbf{W}^t}^T \overline{\nabla f}_{t+1}\right\} + 2\tfrac{||\overline{\nabla f}_{t+1}||_2}{||\mathbf{W}^t||_2}\text{Tr}\left\{{\mathbf{W}^t}^T \mathbf{W}^t\right\}\right]\eta_\ell(t)
\end{aligned} \tag{6}$$

where, the condition for $D(\eta) \geq 0$ in (6) is

$$\eta \geq \max\left\{2\frac{\text{Tr}\left\{{\mathbf{W}^t}^T \overline{\nabla f}_{t+1}\right\} + \frac{||\overline{\nabla f}_{t+1}||_2}{||\mathbf{W}^t||_2}\text{Tr}\left\{{\mathbf{W}^t}^T \mathbf{W}^t\right\}}{\text{Tr}\left\{{\overline{\nabla f}_{t+1}}^T \overline{\nabla f}_{t+1}\right\} - \frac{||\overline{\nabla f}_{t+1}||_2^2}{||\mathbf{W}^t||_2^2}\text{Tr}\left\{{\mathbf{W}^t}^T \mathbf{W}^t\right\}}, 0\right\}. \tag{7}$$

The lower bound in (7) proves the existence of a lower bound for monotonicity condition.

Our final inspection is to check if the substitution of step-size (7) in (5) would still hold the inequality condition in (5). Followed by the substitution, the inequality should satisfy

$$\eta_\ell(t) \geq \zeta \frac{1}{N\gamma} D. \tag{8}$$

We have found that $D(\eta) \geq 0$ for some lower bound in (7), where the inequality in (8) also holds from some $\zeta \geq 0$ and the proof is done.

2. Proof for Proposition 1

Consider the RMSGD update rule for Stochastic Gradient Descent with Momentum

$$\begin{aligned} \boldsymbol{v}_\ell^k &= \alpha v_\ell^{k-1} - \eta_\ell(t)\boldsymbol{g}_\ell^k, \\ \boldsymbol{w}_\ell^{k+1} &= \boldsymbol{w}_\ell^k + \boldsymbol{v}_\ell^k. \end{aligned}$$

It is evident that $\boldsymbol{w}_\ell^k = \boldsymbol{w}_\ell^{k-1} + \boldsymbol{v}_\ell^{k-1}$ and thus, $\boldsymbol{v}_\ell^{k-1} = \boldsymbol{w}_\ell^k - \boldsymbol{w}_\ell^{k-1}$. We can then write our parameter update rule as

$$\boldsymbol{w}_\ell^{k+1} = \boldsymbol{w}_\ell^k + \alpha(\boldsymbol{w}_\ell^k - \boldsymbol{w}_\ell^{k-1}) - \eta_\ell(t)\boldsymbol{g}_\ell^k.$$

Note how for RMSGD, the parameter update rule is subject to the parametrization of $\eta_\ell(t)$ by ℓ (per layer) and t (per epoch).

We highlight the convergence of SGD algorithm is given by the following theorem.

Theorem 1 *(Convergence of Momentum SGD, Theorem 3.2 introduced in [7] (page 132)). Let the following assumptions hold*

1. *The loss function f is convex (Assumption 1.2 in [7], page 128) i.e.*

$$f(\boldsymbol{w}'^k) \geq f(\boldsymbol{w}^k) + < \nabla f(\boldsymbol{w}^k), \boldsymbol{w}'^k - \boldsymbol{w}^k >, \quad \textit{for all } \boldsymbol{w}^k, \boldsymbol{w}'^k \tag{9}$$

2. *The loss function is continuously differentiable as well as the gradient of the loss function is Lipschitz continuous with Liptschitz constant $C > 0$ (Assumption 1.3 in [7], page 128)*

$$\|\nabla f(\boldsymbol{w}^k) - \nabla f(\boldsymbol{w}'^k)\| \leq C\|\boldsymbol{w}^k - \boldsymbol{w}'^k\|, \quad \textit{for all } \boldsymbol{w}^k, \boldsymbol{w}'^k \tag{10}$$

3. *The normalized variance of the gradient of the loss function is bounded by (Assumptions 1.4-1.5 in [7], page 128)*

$$\frac{Var(\hat{\boldsymbol{g}}^k)}{\|\boldsymbol{g}^k\|^2} \leq M, \quad \textit{for all } k \in \mathbb{N} \tag{11}$$

where $\hat{\boldsymbol{g}}^k$ is an unbiased estimate of the gradient of loss function and $M > 0$ is a positive scalar.

Furthermore, the step-size of momentum SGD satisfies

$$\eta \leq \frac{1-\alpha}{L(M+1)}. \tag{12}$$

Then, the convergence of loss function is guaranteed i.e. $\lim_{k\to\infty} E[f(\boldsymbol{w}^k) - f(\boldsymbol{w}^*)] \to 0$.

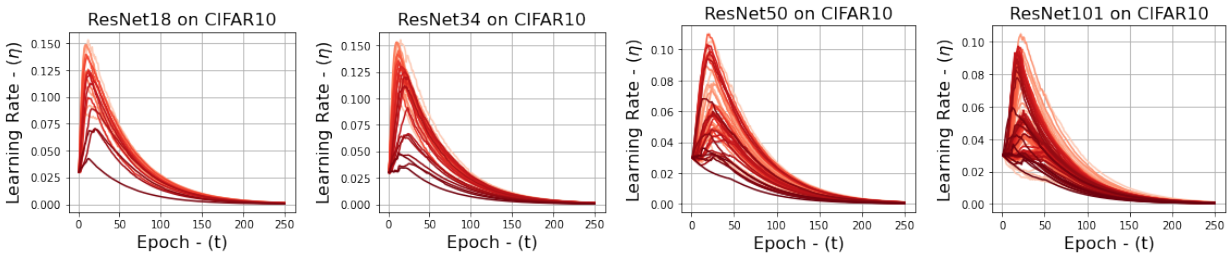
Note the above theorem is deduced from [7] only for the case of momentum SGD where we ignored the Nestrov accelerated version here i.e. $\beta = 0$ in Equation 2.4 in [7], page 130. Given RMSGD, where only the learning rates are evolved given different epoch and layer indexes, then the Liptschitz constant in Equation 10 and the upper bound value in Equation 11 should be revised by the taking the supremum (least upper bound) across all epochs and layers

$$C' \leftarrow \sup_{t,\ell} C_\ell^t \quad \text{and} \quad M' \leftarrow \sup_{t,\ell} M_\ell^t. \tag{13}$$

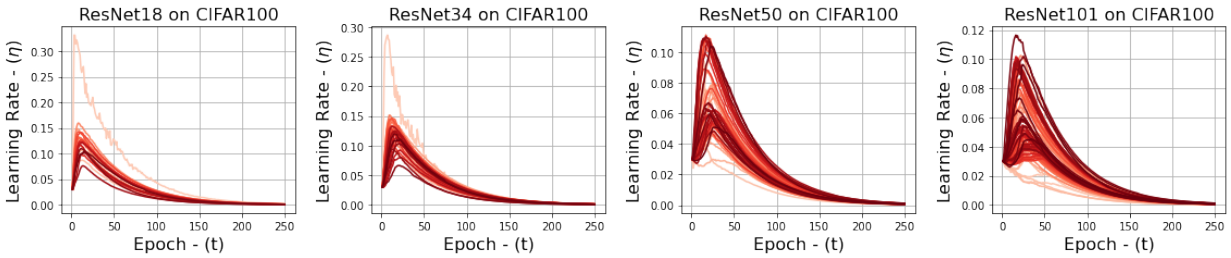
The convergence of RMSGD is now guaranteed by revising the upper bound in Equation 12 to satisfy

$$\eta_\ell(t) \leq \frac{1-\alpha}{C'(M'+1)}. \tag{14}$$

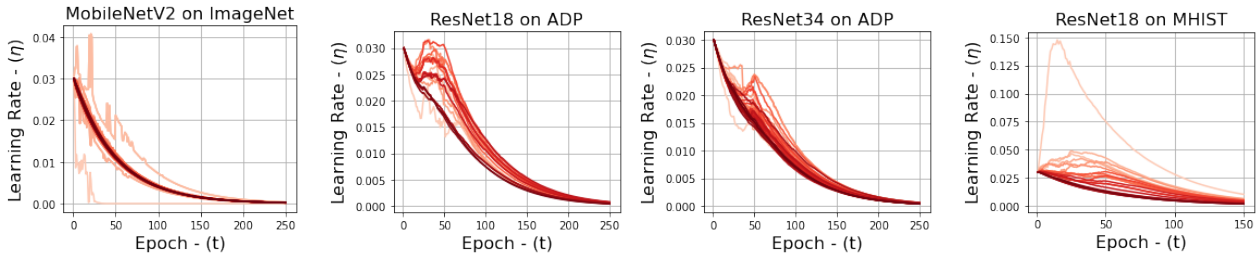
The condition in Equation 14 induces a higher bound on the learning rates across all epochs and layers. This can be generally satisfied assuming the Liptschitz continuity of gradients in Equation 10 and bounded variance of the gradients in Equation 11 hold. We empirically evaluate this in Figure 1 on different scenarios given different dataset and network for training. As it shows, the learning rates are fairly bounded and do not explode in practice.



(a) CIFAR10



(b) CIFAR100



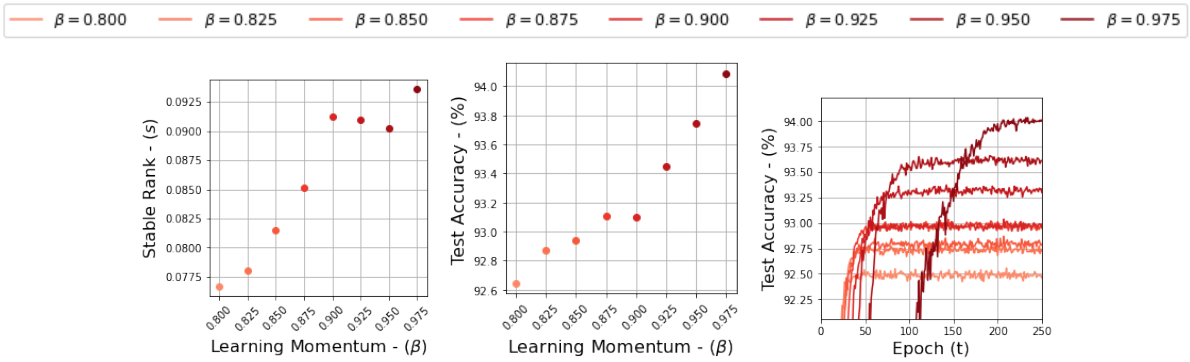
(c) ImageNet (d) ADP (e) MHIST

Figure 1. Learning rate evolution over full 250 of trainig for various ResNets applied on CIFAR10 and CIFAR100, as well as some experiments on ImageNet, ADP, and MHSIT. Increasing darkness of lines indicates increasing layer index for each network. Note that each network starts at $\eta = 0.03$, and we used a batch size of 128 for CIFAR and ImageNet experiments, and 32 for ADP and MHIST. These learning rates are reported as the average learning rate over 5 trials, except for ImageNet, it's over 3 trials.

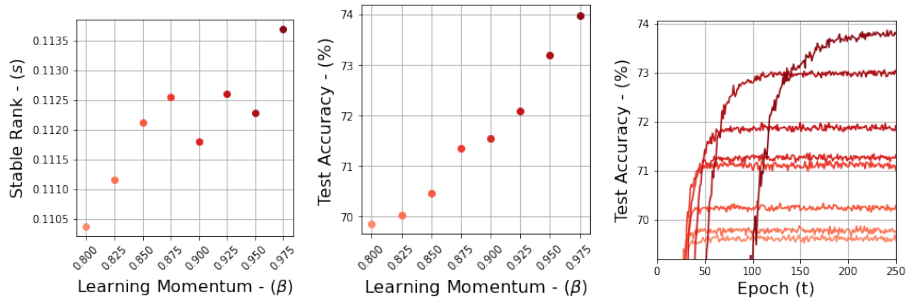
Appendix-B: Additional Experiments, Hyper-Parameter Tuning, and More

3. Learning Momentum (β) Ablative Study

We present the rest of our learning momentum ablative study in Figure 2. Recall that we performed our ablative study using VGG16 on CIFAR10 and CIFAR100, with a batch size of 128. Note that Figure 2 highlights how the β parameter may be used to tradeoff between faster convergence speed (lower β) and better performance (high β) – but at the cost of longer convergence speeds.



(a) CIFAR10



(b) CIFAR100

Figure 2. Learning momentum ablative study for VGG16 applied to CIFAR10 and CIFAR100, with a batch size of 128. We show stable rank behaviour and corresponding test accuracy plots, to visualize the connection between stable rank and resulting performance, since stable rank is utilized in RMSGD's update algorithm. Notice the correlated positive trends with increasing β.

4. A Note on Quality Measure Results in the Main Draft

We provide here some additional details on the experimental details pertaining to Figure 2 in the main draft, which shows our quality measure on CIFAR10/CIFAR100. Specifically, each scatter point represents one of AdaBound, AdaGrad, Adam, AdamP, SAM, SLS, SGD, SGDP, or RMSGD on ResNet18, ResNet34, ResNet50 or ResNet101. Naturally, each permutation of these networks and optimizers are then applied to CIFAR10 and CIFAR100. We used a batch size of 128. For hyper-parameter details, see subsubsection 5.2.1.

5. Experiments

5.1. Hardware

In total, we had a server with 4 RTX2080Ti GPUS, an Intel Xeon Gold 6246 processor, and 256 gigabytes of RAM available for all experiments. All experiments reported here were performed on an instance with access to a single Nvidia RTX2080Ti GPU, 4 cores of the Intel Xeon Gold 6246 processor, and 64 gigabytes of RAM. No experiments were performed using any GPU parallelism.

5.2. Image Classification

We present additional results and experimental details from image classification experiments here.

We perform 5 randomly initialized trials of each experiment except ImageNet, for which we performed 3 trials, and we report the mean and standard deviation of results.

Note on training time. Since different optimizers exhibit widely different epoch times, for a fair comparison, we limit training to the total wall clock time consumed by 250 epochs using SGD. In terms of epochs, this amounts to ~ 250 epochs for all optimizers except SAM, which consumes only $\sim 128 - 133$ epochs due to its 2 forward passes.

5.2.1 Hyper-Parameters

For all image classification tasks, we tuned each optimizer on ResNet18 applied to CIFAR10, with a batch size of 128. We used a step decay method of step size 25 and gamma 0.5 for all optimizers, except RMSGD (who had no learning rate decay). We found this setup to be optimal, as also reported in [6]. We used a consistent weight decay of 5×10^{-4} for all optimizers, momentum rate of 0.9 for SGD, SGDP, SAM, and RMSGD, and kept all other hyper-parameters as default. For learning rates, for RMSGD we consider a grid search over, $\{0.01, 0.02, 0.025, \mathbf{0.03}, 0.035, 0.04, 0.05\}$, with the selected learning rate bold. For AdaBelief, AdamP, SLS, SAM, and SGDP, we follow the CIFAR-specific hyper-parameter search ranges presented in their original work. For Adam, AdaGrad, and RMSProp, we followed the ranges presented in [6] and confirmed in our experiments that they yield optimal performance over the author-suggested values. For SGD, we considered a range of $\{0.05, 0.08, 0.09, \mathbf{0.1}, 0.11, 0.12\}$. We tabulate the final selected learning rate for each method in 1. In general, the selected learning rate results to the suggested learning rate provided in each optimizer's original work. We highlight once again that these learning rates and hyper-parameters were carried forward for all other experiments, including all other networks on CIFAR10/CIFAR100, cutout experiments on CIFAR10/CIFAR100, ImageNet, and both histopathology pathology datasets.

Table 1. Final selected learning rates for each optimizer, tuned using ResNet18 on CIFAR10 using a batch size of 128. We tuned by completing a full 250 epoch training cycle, and selected based on final validation top-1 accuracy.

AdaBound	AdaGrad	Adam	AdamP	SLS	SAM	SGD	SGDP	RMSGD
0.01	0.01	0.0003	0.01	1.0	1.0	0.1	0.1	0.03

5.2.2 Dataset Details

CIFAR. We consider a batch size of 128. For CIFAR-related experiments, we perform 32×32 random-resize cropping and random horizontal flipping as data augmentations.

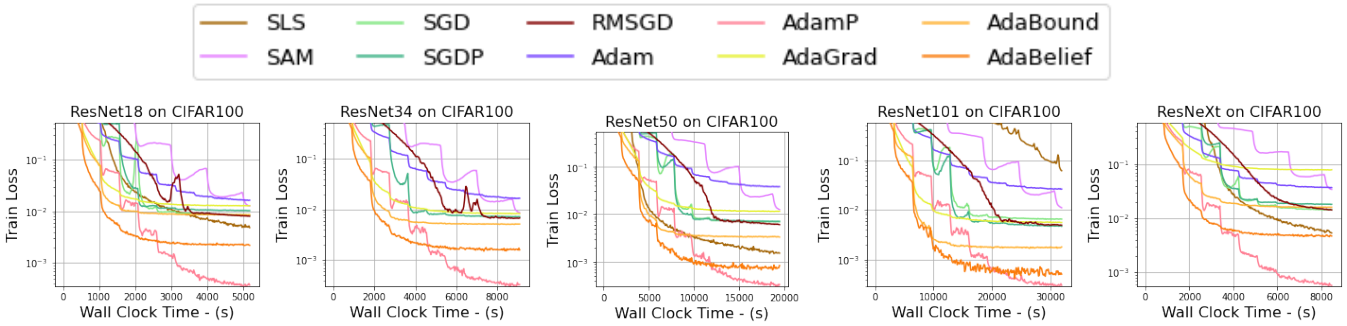
ImageNet. We perform 3 randomly initialized trials and report the mean and standard deviation of results. We consider a batch size of 128, as we performed experiments on a single GPU instance without any GPU parallelism. We follow [2] and perform random resized cropping to 224×244 and random horizontal flipping on the train set and 256×256 resizing with 224×224 center cropping on the test set.

Histopathology Datasets. The histopathlogy datasets provide a study of optimizers on a low class and small sized dataset, with MHIST having $2,175$ training examples across binary class with 224×224 image size, and ADP having $14,134$ examples across 22 multi-labeled classes with 272×272 image size. We consider a batch size of 32 for both datasets following the recommendations in [4, 5]. We consider random cropping 224×224 and random flipping as data augmentations for MHIST, and only random flipping for ADP. See MHIST [5] and ADP [4] references for instructions on how to download the datasets.

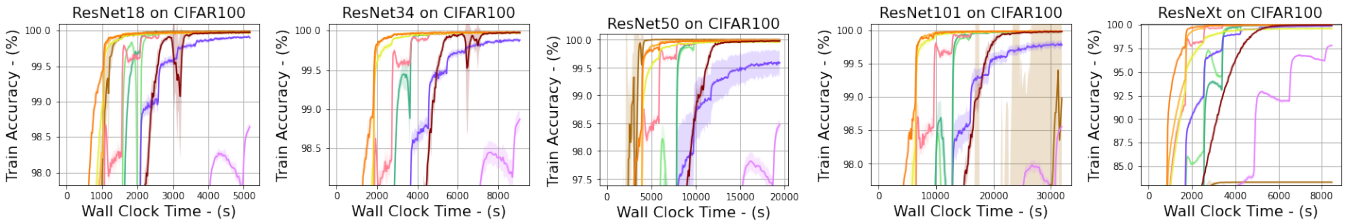
5.2.3 Results

Figures 3, 4, 5, 6, 8, 7 visualize the train loss, test accuracy, and train accuracy for CIFAR experiments. Table 3 shows the final epoch results of CIFAR experiments without cutout, and Table 4 shows the final epoch results of CIFAR experiments with cutout. Note that SAM here has consumed twice as many seconds to train, and unfortunately we do not have the analogous 500-epoch training results for the other optimizer, which makes it an unfair comparison to the other optimizers.

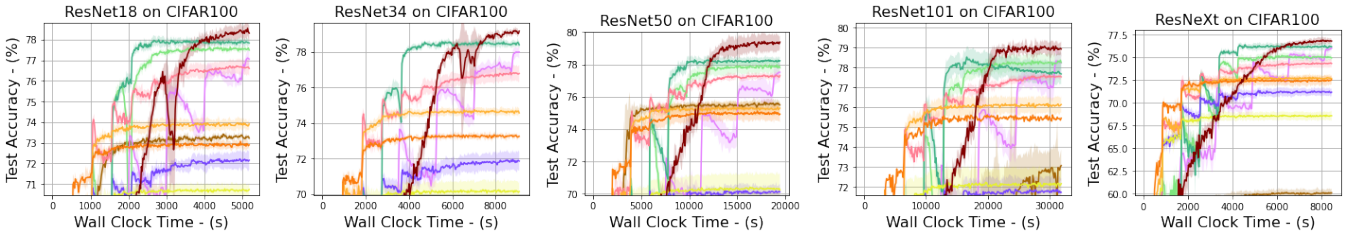
Figure 10 visualizes the test and train accuracy, and the train loss for MHIST experiments. Figure 9 shows the test and train accuracy, and the train loss for ADP experiments. Table 2 tabulates the test area under the curve (AUC) results for MHIST experiments.



(a) Train Loss

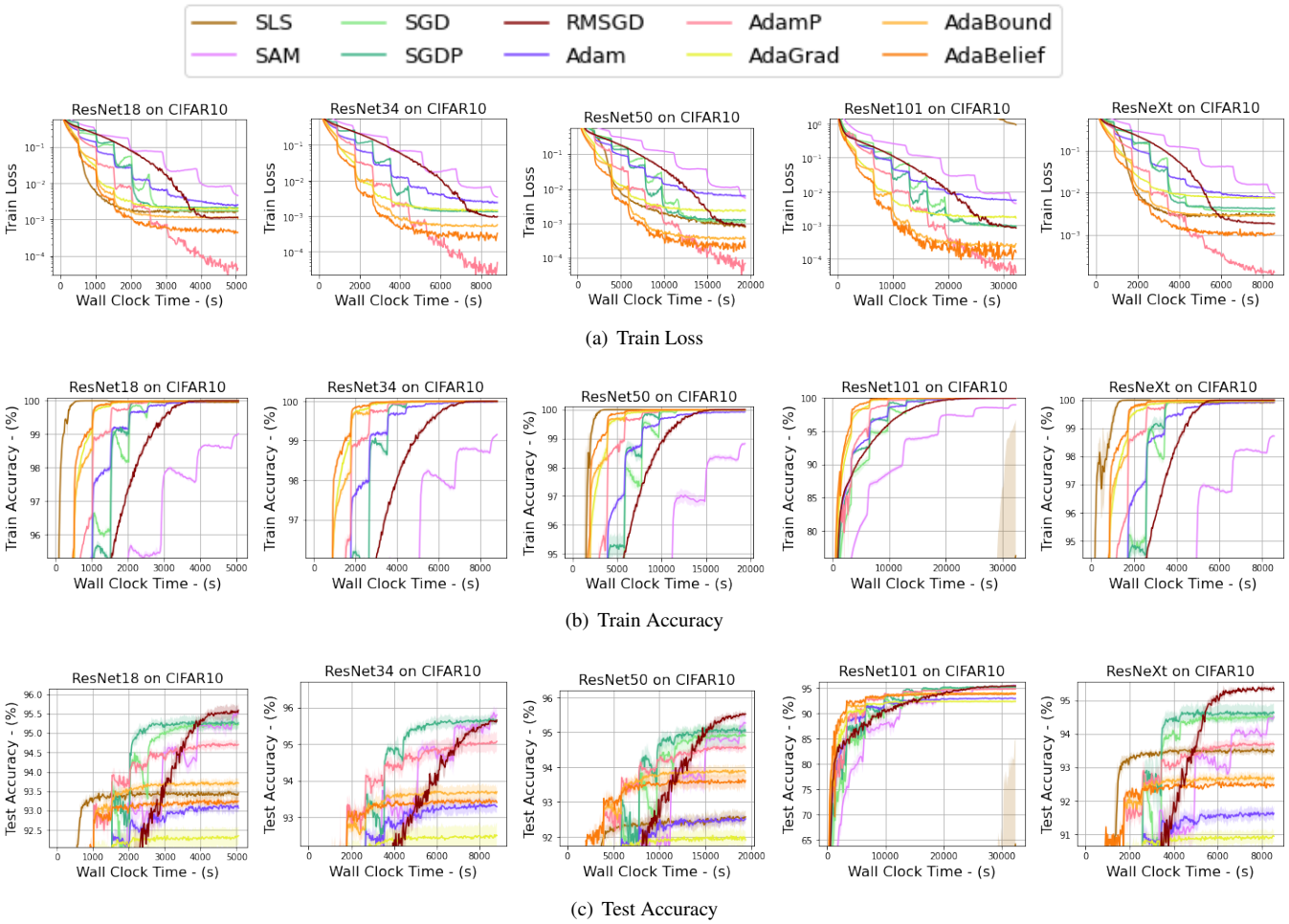


(b) Test Accuracy



(c) Test Accuracy

Figure 3. Train accuracy, test accuracy, and train loss for ResNets and ResNeXt on CIFAR100 experiments, without Cutout. A batch size of 128 was used, and we report the mean over 5 trials for each experiment, with translucent bands to indicate the standard deviation of each experiment. All networks were tuned using ResNet18 applied on CIFAR10.

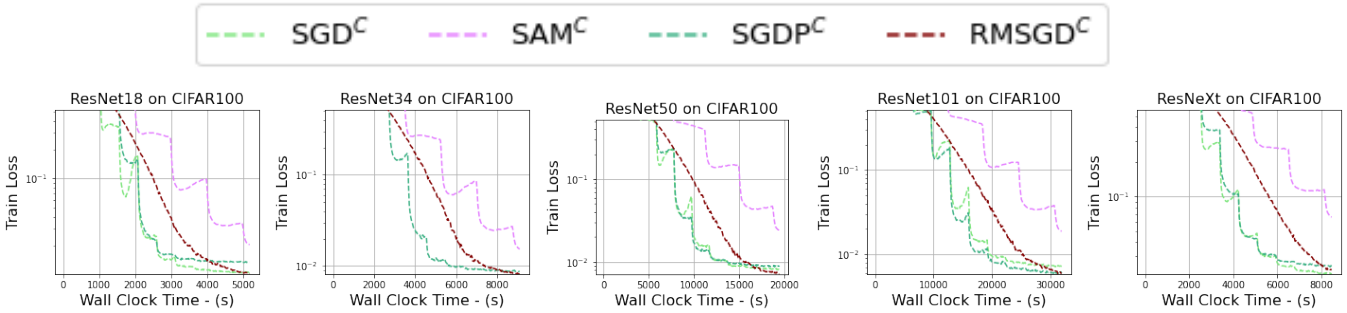


(a) Train Loss

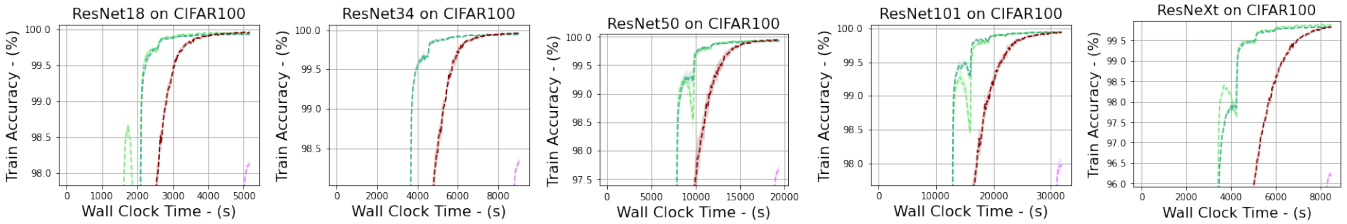
(b) Train Accuracy

(c) Test Accuracy

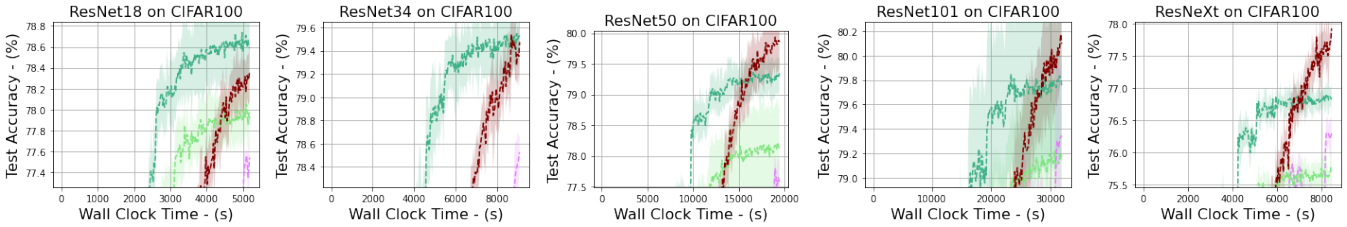
Figure 4. Train accuracy, test accuracy, and train loss for ResNets and ResNeXt on CIFAR10 experiments, without Cutout. A batch size of 128 was used, and we report the mean over 5 trials for each experiment, with translucent bands to indicate the standard deviation of each experiment. All networks were tuned using ResNet18 applied on CIFAR10.



(a) Train Loss

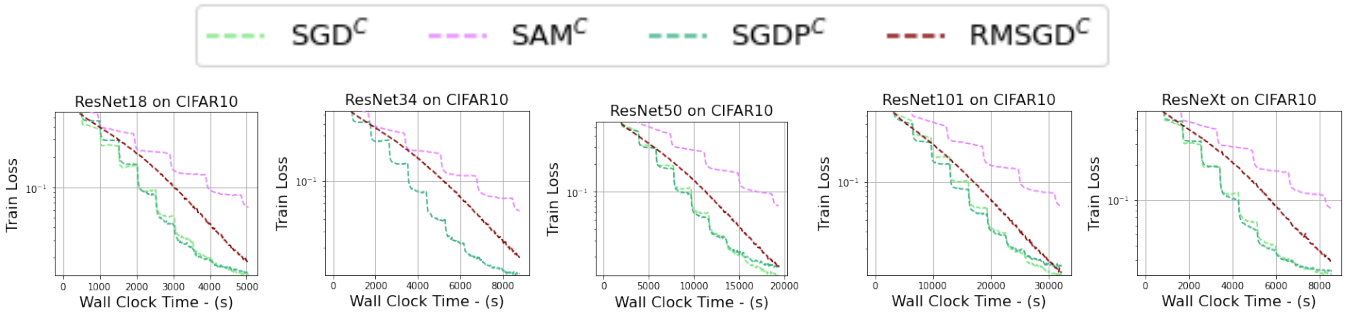


(b) Test Accuracy

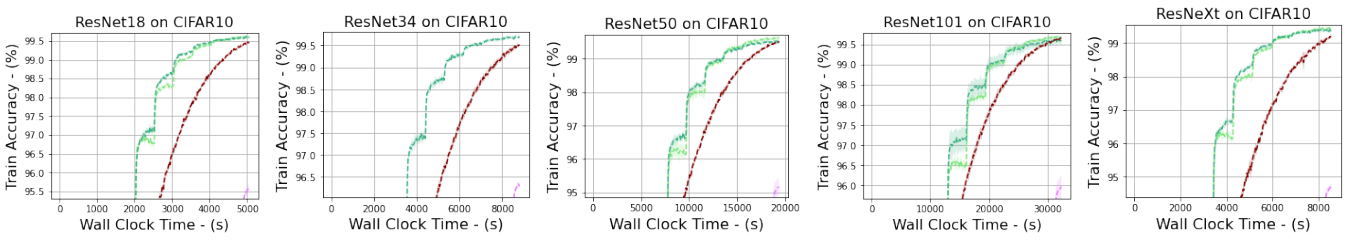


(c) Test Accuracy

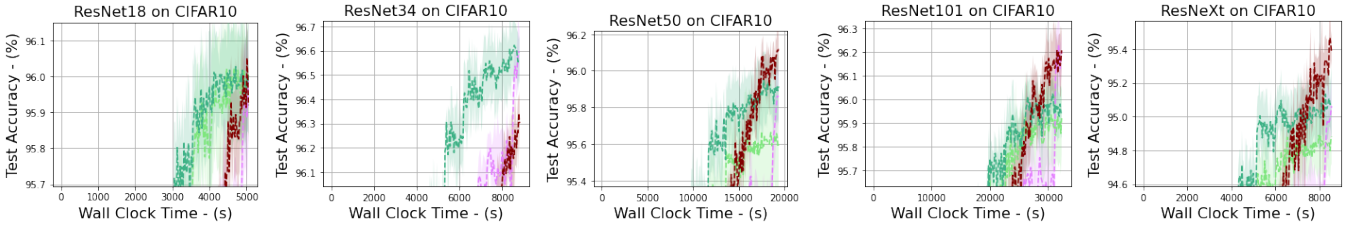
Figure 5. Train accuracy, test accuracy, and train loss for ResNets and ResNeXt on CIFAR100 experiments, with Cutout. A batch size of 128 was used, and we report the mean over 5 trials for each experiment, with translucent bands to indicate the standard deviation of each experiment. All networks were tuned using ResNet18 applied on CIFAR10.



(a) Train Loss

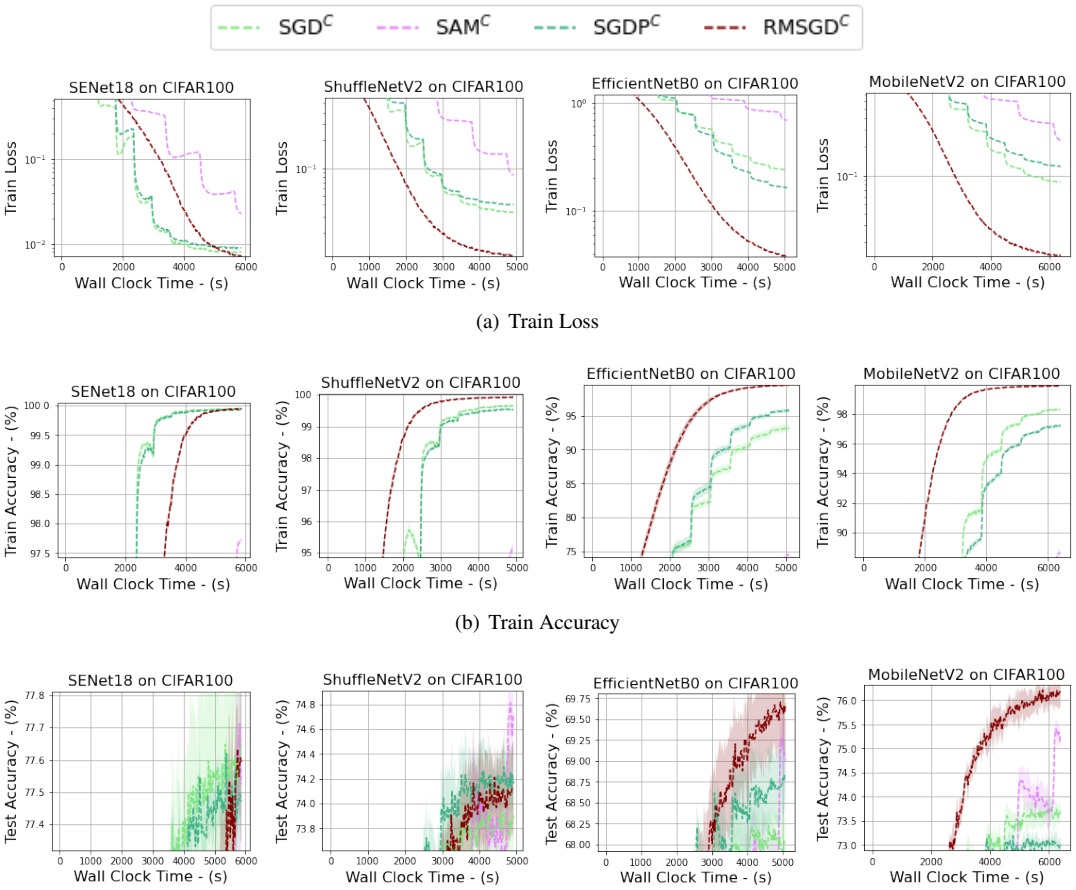


(b) Train Accuracy



(c) Test Accuracy

Figure 6. Train accuracy, test accuracy, and train loss for ResNets and ResNeXt on CIFAR10 experiments, with Cutout. A batch size of 128 was used, and we report the mean over 5 trials for each experiment, with translucent bands to indicate the standard deviation of each experiment. All networks were tuned using ResNet18 applied on CIFAR10.

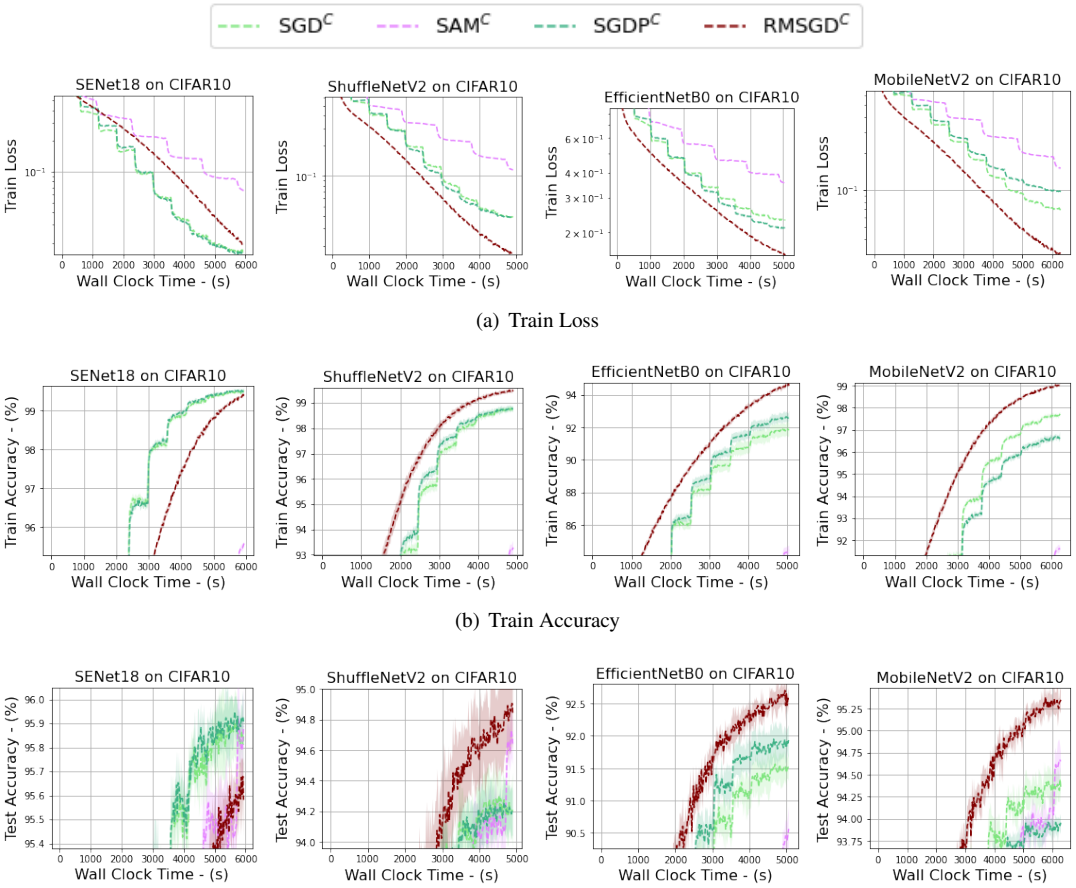


(a) Train Loss

(b) Train Accuracy

(c) Test Accuracy

Figure 7. Train accuracy, test accuracy, and train loss for other networks (SeNet18, ShuffleNetV2, EfficientNetB0, MobileNetv2) on CIFAR100 experiments. These plots include only cutout experiments. A batch size of 128 was used, and we report the mean over 5 trials for each experiment, with translucent bands to indicate the standard deviation of each experiment. All networks were tuned using ResNet18 applied on CIFAR10.

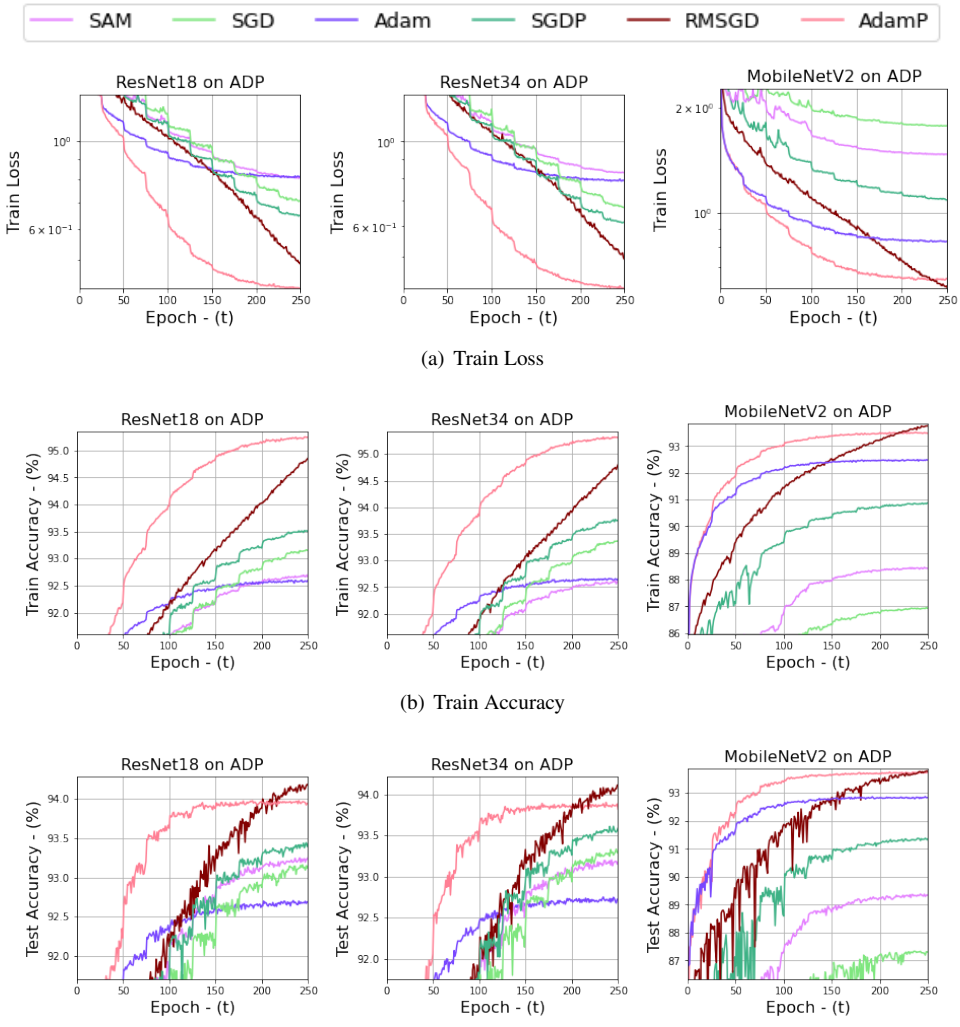


(a) Train Loss

(b) Train Accuracy

(c) Test Accuracy

Figure 8. Train accuracy, test accuracy, and train loss for other networks (SeNet18, ShuffleNetV2, EfficientNetB0, MobileNetv2) on CIFAR10 experiments. These plots include only cutout experiments. A batch size of 128 was used, and we report the mean over 5 trials for each experiment, with translucent bands to indicate the standard deviation of each experiment. All networks were tuned using ResNet18 applied on CIFAR10.

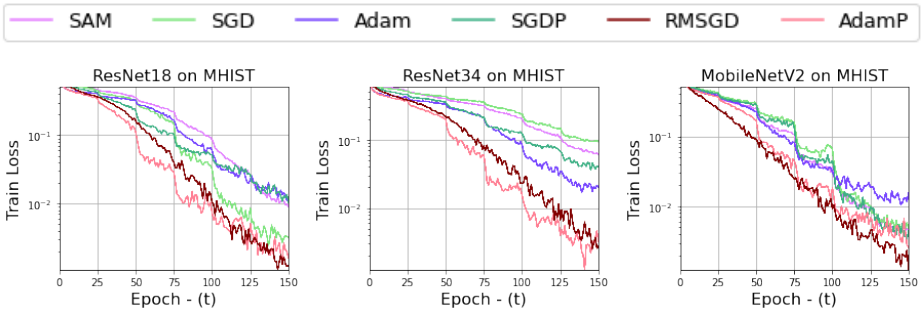


(a) Train Loss

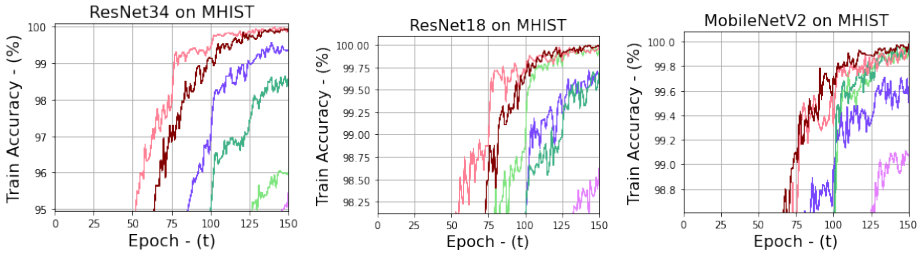
(b) Train Accuracy

(c) Test Accuracy

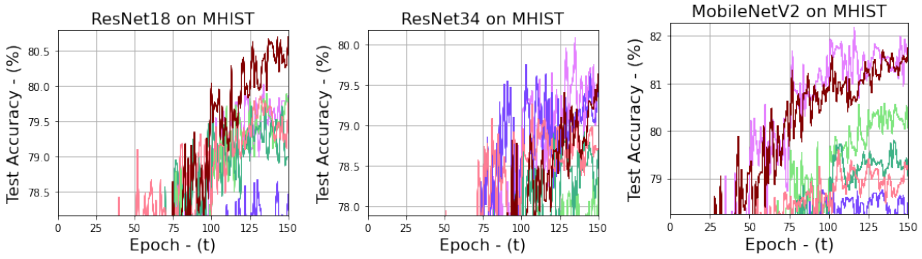
Figure 9. Train accuracy, test accuracy, and train loss for ResNet18, ResNet34, and MobileNetV2 on ADP. Each experiment was run with a batch size of 32, and we report the mean over 5 trials. All networks were tuned using ResNet18 applied on CIFAR10.



(a) Train Loss



(b) Train Accuracy



(c) Test Accuracy

Figure 10. Train accuracy, test accuracy, and train loss for ResNet18, ResNet34, and MobileNetV2 on MHIST. Each experiment was run with a batch size of 32, and we report the mean over 5 trials. All networks were tuned using ResNet18 applied on CIFAR10.

Table 2. Test area under the curve (AUC) results for experiments on MHIST. We reported the mean and standard deviation over 5 trials. We used a batch size of 32.

Network	Adam	AdamP	SAM	SGD	SGDP	RMSGD
ResNet18	$86.77_{0.23}$	$87.33_{1.05}$	$88.26_{1.43}$	$87.04_{1.75}$	$87.08_{0.53}$	$88.51_{0.63}$
ResNet34	$87.90_{0.44}$	$87.50_{0.65}$	$87.79_{1.22}$	$86.57_{1.98}$	$86.47_{0.77}$	$87.32_{0.98}$
MobileNetV2	$86.58_{0.83}$	$86.77_{0.54}$	$89.34_{0.50}$	$87.48_{0.63}$	$87.47_{2.05}$	$88.88_{0.57}$

Table 3. Perfomance of various networks and optimizers on CIFAR10 and CIFAR100 without Cutout. Results reported for 250 epochs of training, rather than wall clock time. Note that SAM consumes twice as much time to train compared to all other optimizers. The best result is highlighted in green, and for RMSGD results, orange highlights when the results lie with the standard deviation from the best. We used a batch size of 128, and all networks were tuned using ResNet18 on CIFAR10.

Dataset	Network	AdaBound	AdaGrad	Adam	AdamP	SLS	SAM	SGD	SGDP	RMSGD
CIFAR10	ResNet18	$93.84_{0.09}$	$92.45_{0.24}$	$93.27_{0.10}$	$94.82_{0.10}$	$93.62_{0.10}$	$\mathbf{95.98_{0.07}}$	$95.32_{0.07}$	$95.39_{0.16}$	$95.66_{0.17}$
	ResNet34	$93.79_{0.19}$	$92.59_{0.30}$	$93.47_{0.18}$	$95.14_{0.25}$	$93.45_{0.16}$	$\mathbf{96.34_{0.16}}$	$95.56_{0.10}$	$95.75_{0.14}$	$95.71_{0.07}$
	ResNet50	$94.00_{0.15}$	$92.12_{0.23}$	$92.67_{0.12}$	$94.69_{0.10}$	$92.70_{0.18}$	$\mathbf{95.80_{0.18}}$	$95.05_{0.28}$	$95.19_{0.15}$	$\mathbf{95.63_{0.05}}$
	ResNet101	$94.17_{0.13}$	$92.51_{0.22}$	$93.13_{0.08}$	$94.92_{0.24}$	$64.20_{20.98}$	$\mathbf{96.07_{0.12}}$	$95.30_{0.13}$	$95.36_{0.04}$	$95.53_{0.14}$
	ResNeXt	$92.83_{0.14}$	$91.09_{0.19}$	$91.78_{0.16}$	$93.82_{0.10}$	$93.67_{0.09}$	$95.01_{0.09}$	$94.62_{0.09}$	$94.79_{0.24}$	$\mathbf{95.49_{0.05}}$
CIFAR100	ResNet18	$74.09_{0.27}$	$70.92_{0.31}$	$72.45_{0.34}$	$76.81_{0.31}$	$73.59_{0.04}$	$77.82_{0.25}$	$77.80_{0.07}$	$\mathbf{78.13_{0.16}}$	$\mathbf{78.63_{0.34}}$
	ResNet34	$74.84_{0.18}$	$70.39_{0.57}$	$72.09_{0.50}$	$76.93_{0.40}$	$73.22_{0.11}$	$\mathbf{79.22_{0.39}}$	$77.88_{0.39}$	$78.74_{0.12}$	$\mathbf{79.32_{0.10}}$
	ResNet50	$75.52_{0.37}$	$70.60_{0.91}$	$70.53_{0.36}$	$77.47_{0.16}$	$75.80_{0.23}$	$\mathbf{78.85_{0.66}}$	$78.12_{0.42}$	$78.44_{0.24}$	$\mathbf{79.59_{0.54}}$
	ResNet101	$76.31_{0.41}$	$72.39_{0.84}$	$72.20_{0.68}$	$77.71_{0.16}$	$73.31_{0.84}$	$\mathbf{79.71_{0.48}}$	$78.48_{0.45}$	$78.60_{0.55}$	$\mathbf{79.36_{0.26}}$
	ResNeXt	$72.97_{0.38}$	$68.83_{0.43}$	$71.54_{0.41}$	$74.54_{0.40}$	$72.35_{0.42}$	$\mathbf{76.82_{0.30}}$	$75.36_{0.33}$	$\mathbf{76.56_{0.33}}$	$\mathbf{77.14_{0.31}}$

Table 4. Perfomance of various networks and optimizers on CIFAR10 and CIFAR100 with Cutout. Results reported for 250 epochs of training, rather than wall clock time. Note that SAM consumes twice as much time to train compared to all other optimizers. The best result is highlighted in green, and for RMSGD results, orange highlights when the results lie with the standard deviation from the best. We used a batch size of 128, and all networks were tuned using ResNet18 on CIFAR10.

	CIFAR10				CIFAR100			
Network	SAM^C	SGD^C	SGDP^C	RMSGD^C	SAM^C	SGD^C	SGDP^C	RMSGD^C
ResNet18	$\mathbf{96.44_{0.13}}$	$96.12_{0.13}$	$96.13_{0.13}$	$96.13_{0.08}$	$\mathbf{78.77_{0.11}}$	$78.16_{0.21}$	$\mathbf{78.82_{0.37}}$	$\mathbf{78.53_{0.22}}$
ResNet34	$\mathbf{97.10_{0.09}}$	$96.53_{0.13}$	$96.70_{0.10}$	$96.42_{0.08}$	$\mathbf{79.85_{0.19}}$	$78.63_{0.55}$	$\mathbf{79.67_{0.24}}$	$\mathbf{79.70_{0.19}}$
ResNet50	$\mathbf{96.49_{0.10}}$	$95.78_{0.27}$	$96.03_{0.16}$	$96.28_{0.07}$	$79.26_{0.28}$	$78.36_{0.67}$	$\mathbf{79.52_{0.31}}$	$\mathbf{80.06_{0.45}}$
ResNet101	$\mathbf{96.79_{0.08}}$	$96.04_{0.16}$	$96.12_{0.05}$	$96.33_{0.08}$	$\mathbf{80.82_{0.66}}$	$79.35_{0.62}$	$\mathbf{80.03_{0.67}}$	$\mathbf{80.36_{0.35}}$
ResNeXt	$\mathbf{95.67_{0.18}}$	$95.04_{0.18}$	$95.24_{0.15}$	$\mathbf{95.62_{0.08}}$	$77.41_{0.06}$	$75.91_{0.19}$	$77.14_{0.21}$	$\mathbf{78.06_{0.28}}$
MobileNetV2	$\mathbf{95.60_{0.12}}$	$94.53_{0.16}$	$94.07_{0.07}$	$\mathbf{95.48_{0.11}}$	$\mathbf{76.53_{0.18}}$	$73.94_{0.21}$	$73.40_{0.07}$	$\mathbf{76.36_{0.27}}$
SENet18	$\mathbf{96.44_{0.16}}$	$95.99_{0.10}$	$96.04_{0.08}$	$95.80_{0.06}$	$\mathbf{78.85_{0.32}}$	$77.80_{0.23}$	$77.70_{0.05}$	$77.77_{0.15}$
EfficientNetB0	$91.83_{0.23}$	$91.70_{0.24}$	$92.09_{0.21}$	$\mathbf{92.83_{0.14}}$	$\mathbf{70.47_{0.50}}$	$68.42_{0.24}$	$68.98_{0.44}$	$\mathbf{69.87_{0.49}}$
ShuffleNetV2	$\mathbf{95.39_{0.15}}$	$94.40_{0.21}$	$94.37_{0.12}$	$\mathbf{95.01_{0.29}}$	$\mathbf{75.88_{0.19}}$	$74.13_{0.35}$	$74.44_{0.29}$	$74.38_{0.36}$

5.3. Epoch Times

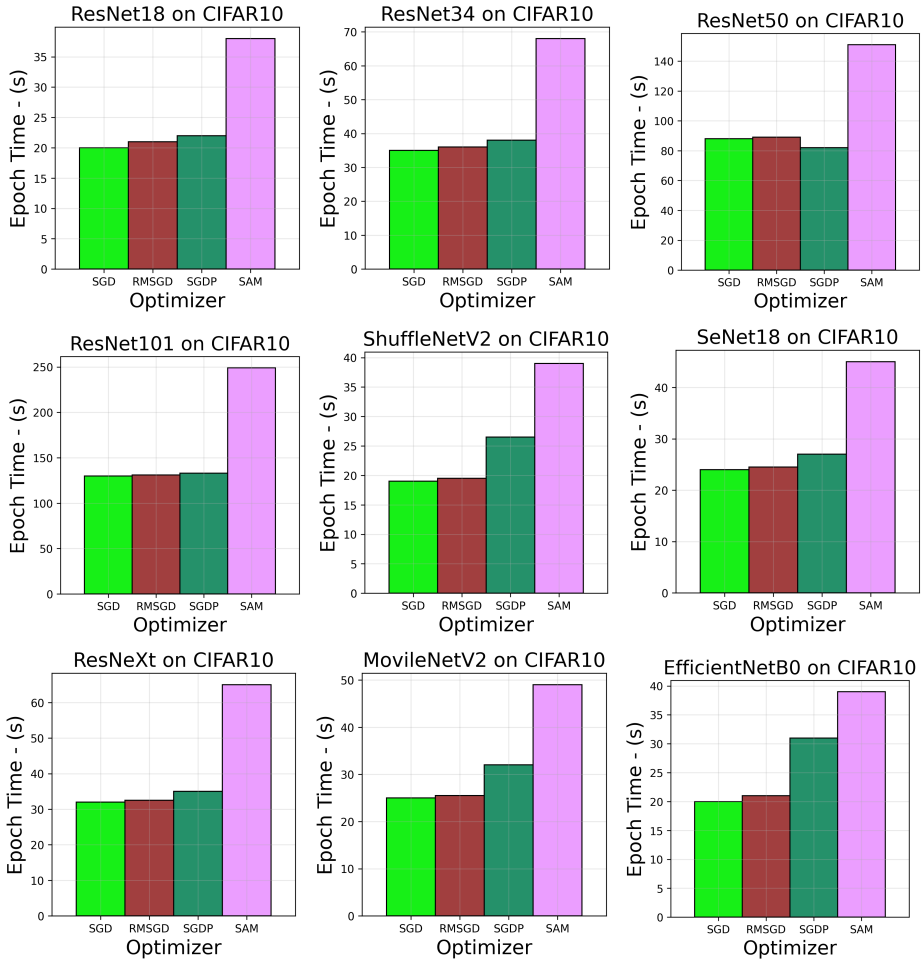


Figure 11. Epoch times for various networks on CIFAR10 using SDG, RMSGD, SGDP, and SAM.

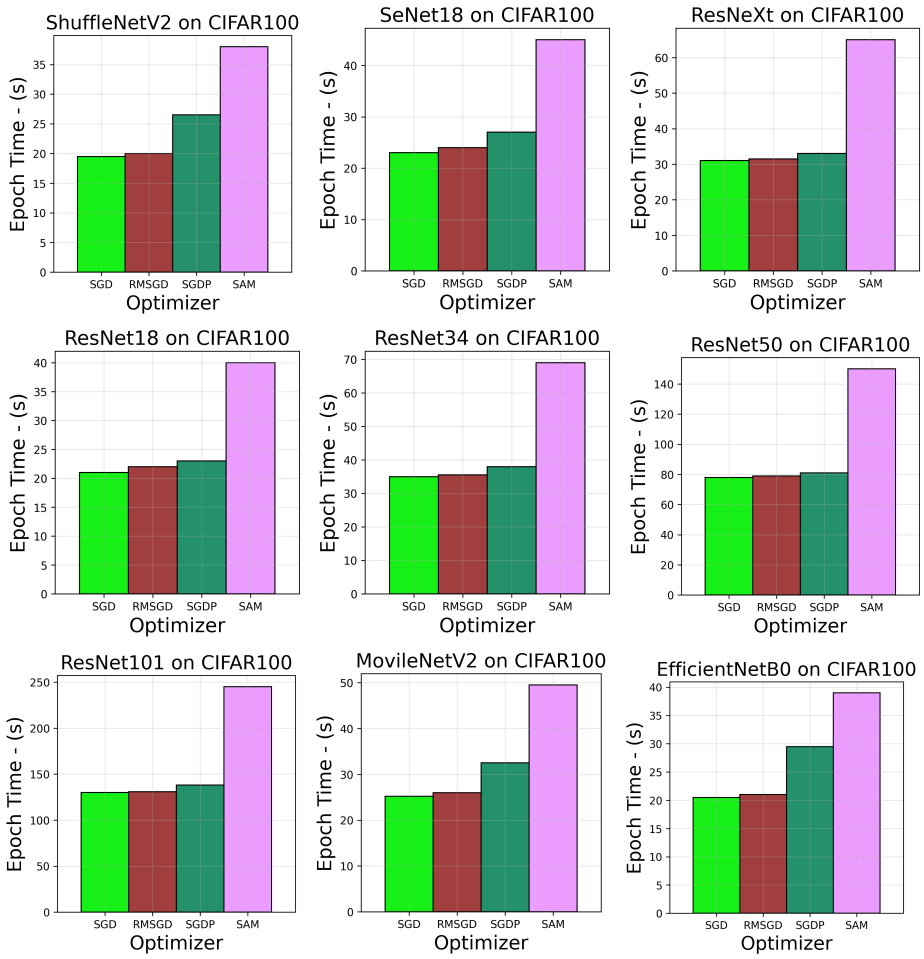


Figure 12. Epoch times for various networks on CIFAR100 using SDG, RMSGD, SGDP, and SAM.

5.4. Language Modelling

5.4.1 General Notes

The Penn TreeBank dataset is a word-level language modelling dataset. We first perform comparisons to Adam, SGD, AdamW, RAdam, and AdaBelief on the 3-layer LSTM, and then only Adam, SGD, and AdaBelief on the 1- and 2-layer LSTM as these optimizers proved most competitive. The model architecture is a 3-layer LSTM. The emebdding size is 400 and the number of hidden units per layer is 1150.

5.4.2 Hyper-Parameters

Note we tuned hyper-parameters on the 3-layer LSTM and then applied the same configuration to the 1- and 2-layer LSTM. For each optimizer, a step decay scheduling of step size 75 epochs and decay rate 0.1 was used. Each optimizer had fixed weight decay of 1×10^{-6}. For SGD and RMSGD, momentum of 0.9 was used. Note for RMSGD, we used $\beta = 0.9$ due to the smaller max epoch number of 200. A batch size of 80 was used, with gradient clipping of 0.25. Each optimizer's learning rate was tuned using a grid search method on the 3-layer LSTM, and the same hyper-parameters were applied to the 1- and 2-layer LSTM. The grids are as follows, with the selected learning rate in bold:

- RMSGD: $\{0.1, 2, 5, 10, \mathbf{15}, 30\}$
- SGD: $\{0.1, 2, 5, 10, 15, \mathbf{30}\}$
- SGDP: $\{0.1, 2, 5, 10, 15, \mathbf{30}\}$
- SAM: N/A
- AdaBelief: $\{0.00005, 0.0001, 0.0003, 0.001, \mathbf{0.01}, 0.1\}$
- AdamW: $\{0.00005, 0.0001, 0.0003, \mathbf{0.001}, 0.010.1\}$
- RAdam: $\{0.00005, 0.0001, 0.0003, \mathbf{0.001}, 0.010.1\}$
- Adam: $\{0.00005, 0.0001, 0.0003, 0.001, \mathbf{0.01}, 0.1\}$

5.5. Generative Adversarial Network

As noted in the main draft, we used the popular Wasserstein GAN (WGAN) [1].

5.5.1 Hyper-Parameters

All hyper-parameters were left as their default values, except for momentum in SGD, RMSGD, and SGDP, which used a momentum rate of 0.9. Note for RMSGD, we used $\beta = 0.95$ due to the smaller max epoch number of 100. Learning rate was the only hyper-parameter we tuned, and we performed a learning rate grid search over the full 100 epochs of training. The learning rate grids are as follows, with the selected learning rate in bold:

- RMSGD: $\{0.001, 0.01, 0.03, 0.1, \mathbf{0.5}, 2\}$
- SGD: $\{0.001, 0.01, 0.03, \mathbf{0.1}, 0.5, 2\}$
- SAM: $\{0.0001, 0.001, 0.01, 0.1, \mathbf{0.5}, 1.0\}$
- AdaBelief: $\{0.00005, 0.0001, \mathbf{0.0003}, 0.001, 0.01, 0.1\}$
- RAdam: $\{0.00005, 0.0001, \mathbf{0.0003}, 0.001, 0.01, 0.1\}$
- Adam: $\{0.00005, 0.0001, \mathbf{0.0003}, 0.001, 0.01, 0.1\}$
- RMSProp: $\{0.00005, 0.0001, \mathbf{0.0003}, 0.001, 0.01, 0.1\}$
- AdaBound: $\{0.00005, 0.0001, \mathbf{0.0003}, 0.001, 0.01, 0.1\}$

References

[1] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan, 2017. 18

[2] 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*, pages 770–778, 2016. 6

[3] Roger A Horn and Charles R Johnson. *Matrix analysis*. Cambridge university press, 2012. 2

[4] Mahdi S Hosseini, Lyndon Chan, Gabriel Tse, Michael Tang, Jun Deng, Sajad Norouzi, Corwyn Rowsell, Konstantinos N Plataniotis, and Savvas Damaskinos. Atlas of digital pathology: A generalized hierarchical histological tissue type-annotated database for deep learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11747–11756, 2019. 6

[5] Jerry Wei, Arief Suriawinata, Bing Ren, Xiaoying Liu, Mikhail Lisovsky, Louis Vaickus, Charles Brown, Michael Baker, Naofumi Tomita, Lorenzo Torresani, et al. A petri dish for histopathology image analysis. In *International Conference on Artificial Intelligence in Medicine*, pages 11–24. Springer, 2021. 6

[6] 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*, pages 4148–4158, 2017. 6

[7] Bai-cun Zhou, Cong-ying Han, et al. Convergence of stochastic gradient descent in deep neural network. *Acta Mathematicae Applicatae Sinica, English Series*, 37(1):126–136, 2021. 3