Model Generalization: A Sharpness Aware Optimization Perspective

Jozef Marus Coldenhoff*
School of Computer and Communication Sciences
EPFL

Chengkun Li*
School of Engineering
EPFL

Yurui Zhu*
College of Humanities
EPFL

Abstract—Sharpness-Aware Minimization (SAM) and adaptive sharpness-aware minimization (ASAM) aim to improve the model generalization. And in this project, we proposed three experiments to valid their generalization from the sharpness aware perspective. And our experiments show that sharpness aware-based optimization techniques could help to provide models with strong generalization ability. Our experiments also show that ASAM could improve the generalization performance on un-normalized data, but further research is needed to confirm this.

Index Terms—Machine Learning, Optimization, Loss sharpness, SAM, ASAM.

I. INTRODUCTION

In recent years, the generalization of deep neural networks has steadily risen to prominence as an important topic in modern machine learning, plenty of works have been done to address the limitations of pure optimization. Better model generalization means more stable results when dealing with unobserved data in real-world applications. Empirical evidence has shown that the shape of the loss function affects how well the models generalize; convergence at a flatter minima is more likely to lead to better generalization.

Stochastic gradient descent (SGD) is known for finding a flat minima that tends to generalize well. Xie and Sato *et al.* [1] theoretically proved that SGD favors flat minima exponentially due to the Hessian-dependent covariance of stochastic gradient noise. Furthermore, Jastrzebski *et al.* [2] theoretically and empirically showed that using a large learning rate and/or small batch size steers SGD towards flatter minima.

Foret and Kleiner *et al.* [3] introduced Sharpness-Aware Minimization (SAM) to improve generalization by simultaneously minimizing both loss value and loss sharpness. Subsequently, Kwon and Kim *et al.* [4] proposed adaptive sharpness-aware minimization (ASAM), which adaptively adjusts maximization regions thus acting uniformly under parameter re-scaling. Since the advent of SAM, many studies have demonstrated that SAM combined with different optimization methods can all improve the generalization. [5], [6], [7], [8]. However, there are not many experiments to validate the performance of ASAM.

In this project, we aim to evaluate how the SAM and ASAM optimizer perform on a Computer Vision image classification task, and whether SGD using SAM and ASAM can truly beat vanilla SGD in terms of generalization performance.

II. METHODS

The Sharpness-Aware Minimization (SAM) performs one more step of gradient ascent to approximately determine the worst-case weight perturbation before updating the weights [3]. This is done, because in the paper they derive that adding a sharpness minimization objective to the loss function can be optimized by evaluating the

*These authors contributed equally to this work

gradient or the loss at a point in parameter space within an ϵ-ball around the current parameters, and adding that evaluated gradient to the current parameters. This has as an intended effect that the network balances lowering the training loss, and its sharpness. More specifically, [3] defines the sharpness of loss function as equation 1.

$$\max_{\|\boldsymbol{\epsilon}\|2 \leq \rho} L_S(\mathbf{w} + \boldsymbol{\epsilon}) - L_S(\mathbf{w}) \tag{1}$$

And the sharpness-aware minimization can be defined as the following min-max optimization:

$$\min \mathbf{w} \max_{\|\boldsymbol{\epsilon}\|p \leq \rho} L_S(\mathbf{w} + \boldsymbol{\epsilon}) + \frac{\lambda}{2}\|\mathbf{w}\|_2^2 \tag{2}$$

And the weights can be updated after the ϵ is calculated, using equation 3:

$$\begin{cases} \boldsymbol{\epsilon}_t = \rho \frac{\nabla LS(\mathbf{w}_t)}{\|\nabla LS(\mathbf{w}_t)\|2} \\ \mathbf{w}_{t+1} = \mathbf{w}_t - \alpha_t \left(\nabla L_S \left(\mathbf{w}_t + \boldsymbol{\epsilon}_t \right) + \lambda \mathbf{w}_t \right) \end{cases} \tag{3}$$

However, [4] note that as was shown by [9] a rectifier neural network can be arbitrarily reparameterized in such a way that the function of the network stays the same, while changing the sharpness of the loss landscape. [4] then suggest that rescaling the ϵ-ball to match the scale of the parameters could improve the performance of the SAM algorithm by removing the scale-dependence from the loss maximization step. This resulted in the ASAM optimizer, where the maximization problem is changed to:

$$\max_{\|\boldsymbol{T_w^{-1}}\boldsymbol{\epsilon}\|2 \leq \rho} L_S(\mathbf{w} + \boldsymbol{\epsilon}) - L_S(\mathbf{w}) \tag{4}$$

Where T_w^{-1} is a rescaling operator defined by the magnitudes of the parameters in fully connected and convolution layers.

The final weight update is the calculated as:

$$\begin{cases} \boldsymbol{\epsilon}_t = \rho \frac{T_w^2 \nabla LB(\mathbf{w}_t)}{\|T_w \nabla LS(\mathbf{w}_t)\|2} \\ \mathbf{w}_{t+1} = \mathbf{w}_t - \alpha \left(\nabla L_B \left(\mathbf{w}_t + \boldsymbol{\epsilon} \right) + \lambda \mathbf{w}_t \right) \end{cases} \tag{5}$$

We carried out the following experiments to discover model's generalization from the sharpness aware perspective.

III. EXPERIMENTS

A. Experiment A

The aim of our first experiment is to test the generalization ability of model trained with *Sharpness Aware Minimization* (SAM) and vanilla *Stochastic Gradient Descent* (SGD) on image classification task at testing time. To this end, we trained ResNet-18 on CIFAR-10 for 100 epochs using SAM and SGD. For SAM, it needs a base optimizer, so we use a SGD with the same configuration with the SGD we compared to in order to ensure fairness. The SGD is trained with 0 momentum and constant learning rates of 0.01 (the learning

rate is selected by grid search with the goal of balancing training time and training stability) with batch size of 128, for image classification task we use Cross Entropy Loss. We measured the classification accuracy of models trained with these two optimizers on test data-set to determine their generalization ability.

Additionally, since SAM updates twice per training step, so we compare the testing accuracy of SAM at first 50 epochs with the testing accuracy of SGD at first 100 epochs. We report the results in Table I which are measured as average across 3 seeds.

Seed	Method	Testing accuracy
20	SAM	**71.22**
	SGD	65.15
30	SAM	**70.77**
	SGD	64.03
40	SAM	**70.98**
	SGD	64.17

TABLE I
RESULT OF EXPERIMENT 1: COMPARING THE TESTING ACCURACY OF MODELS TRAINED WITH SGD/SAM.

From the result we can see that with every random initialization, SAM has a significantly higher testing accuracy than vanilla SGD.

B. Experiment B

According to Andriushchenko *et al.* [5], using smaller batch sizes steers SGD toward flatter minima, and in contrast, a large batch size theoretically makes it harder for SGD harder to escape from sharper minima due to less noise in the gradient. In the second experiment, we will explore whether SAM helps to mitigate the poor generalization associated with larger batch sizes of SGD and see if it can help generalize better in the test time by finding a flatter minimum. We here trained the ResNet-18 on CIFAR-10 for 100 epochs using both vanilla SGD and SAM with the same configuration in Section III-A.

We report the testing accuracy w.r.t training epochs and training loss (in log scale) w.r.t training epochs in Figure 1 and Figure 2. All curves are averaged across 3 seeds with standard deviation plotted in light colors.
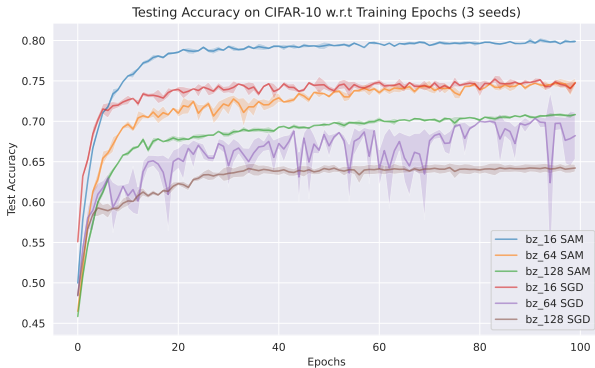

Fig. 1. Testing accuracy on CIFAR-10 with SAM/SGD trained with different batch sizes and initial weights.

Testing Accuracy Interpretation: From Figure 1, we can observe that models trained with smaller batch size generalize better at testing time for models trained with either SAM or SGD, which is in consistent with the observations in [5]. However, from our experiments, we observe that the test accuracy of the models using SGD varies more than the test accuracy of the models using SAM across 3 seeds, this is especially evident in larger batch sizes (64 and 128), where the variance of the SAM-based model is significantly

lower than that of the SGD-based model. In addition, comparing the models trained with SAM/SGD for the same batch size, the models trained with SAM achieved test accuracy at around $5 \sim 10\%$ higher than SGD.
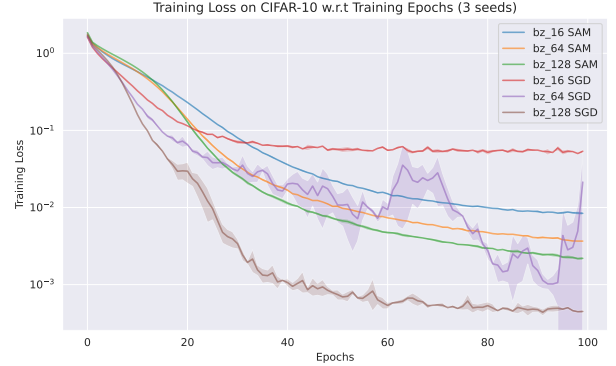

Fig. 2. Training loss on CIFAR-10 with SAM/SGD trained with different batch sizes and initial weights.

Training Loss Interpretation: From Figure 2 we can easily distinguish the loss curves of the SAM-based models from the SGD-based models by either the convergence rate or the variance of loss. Interestingly, our set of experiments shows that lower losses do not necessarily lead to higher accuracy in test time, and in addition, we can see that faster convergence in the case of SGD during training represents lower generalization performance in our experiments as well. On the other hand, we can observe a similar phenomenon for the loss curve variance, i.e., the SGD-based model has a larger variance in the loss curve compared to the SAM-based model. We interpret this result in the perspective of sharpness, this high variance may come from different weight initializations, where training with SGD may be stuck at different sharp local minima, while training with SAM leads to flatter minima, or perhaps even the same minima for different weight initializations.

C. Experiment C

ASAM intends to address the parameter scaling problem that SAM and other sharpness minimization-based optimizers may encounter. As it has been shown by [9] that one may arbitrarily reparameterize rectifier networks in such a way that the function of the network stays the same, while changing the flatness of the loss landscape. ASAM tackles this by rescaling the ϵ-ball where the loss is maximized to reflect the different magnitudes of the parameters.

We wanted to see how much this rescaling problem affects SAM in an image classification task and whether ASAM can solve the problem. Hence we conduct an experiment to test this. We design the experiment to train ResNet-18 on CIFAR-10 for 200 epochs using vanilla SGD with SAM and ASAM, in order to find whether ASAM is more robust to parameter scaling, we remove all Batch Normalization layers [10]. We conduct this experiment with the same hyperparameters as the ASAM authors use in their comparison between SAM and ASAM using CIFAR-10 and ResNet 20. The learning rate for both SAM and ASAM is set to 0.01, batch size of 128, momentum and weight decay are set to 0.9 and 0.0005, and finally, a cosine annealing learning rate scheduler [11] is used to train the networks. For this experiment, we use the standard data normalization of CIFAR-10, where the input pixels have an intensity between 0 and 1. For both SAM and ASAM, we use three seeds to initialize the network weights and average the results in the end.
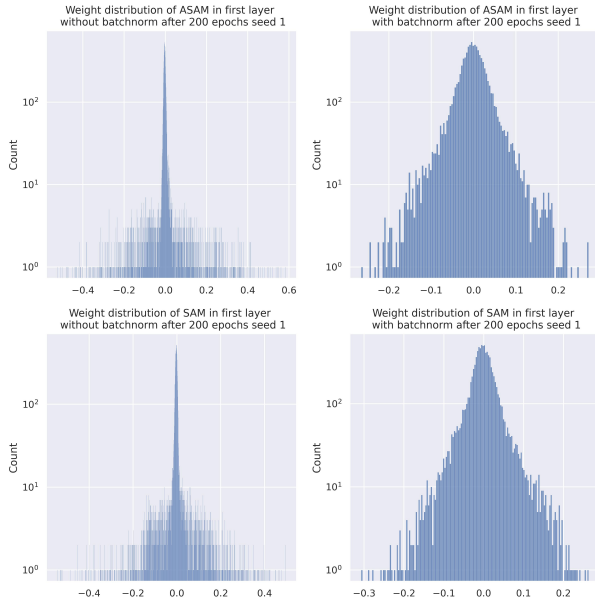
Fig. 3. Weight distribution for ASAM and SAM in the first convolution layer.

Seed	Method	Testing accuracy	Training loss (10^{-5})
1	SAM	**70.06**	**8.72**
	ASAM	68.65	27.5
2	SAM	65.44	**7.19**
	ASAM	**65.77**	25.0
3	SAM	65.49	**9.50**
	ASAM	**66.23**	24.7

TABLE II

TABLE SHOWING THE RESULTS OF SAM AND ASAM TRAINED ON CIFAR-10

Table II shows the results of the three training runs on both SAM and ASAM. We can observe that in seed 2 and 3, the generalization performance of ASAM is slightly better than that of SAM. However, we see that the training loss of the networks trained with SAM is an order of magnitude lower than that of ASAM. It thus seems that ASAM found a local minima with a higher training loss, while being similar to SAM in terms of generalization.

In order to verify that the removal of the Batch Normalization really led to the result that weights distributing themselves over a larger range, we visualize the distribution of the weights in the first layer of the network when training with or without Batch Normalization for both SAM and ASAM. Figure 3 shows the distribution of the weights with the y-axis in log scale. It seems to show that the removal Batch Normalization created a more heavy tailed distribution, indicating that the weights are of different scale.

Similarly to earlier experiments, we plot how the accuracy and loss evolve over epochs in Appendix 5 and Figure 4 respectively. Figure 4 shows why we observed a difference in the training loss between SAM and ASAM. We observe that in the late stages of training, the SAM optimizer seems to descend further than the ASAM, while keeping similar testing accuracy.

IV. CONCLUSION AND FUTURE WORK

As we have seen in experiment a, using sharpness aware minimization in during training helps to achieve better generalization performance during testing.

In addition, experiment b shows that even though SAM and vanilla SGD both suffer from degradation of generalization performance

Fig. 4. Training loss on CIFAR-10 with SAM/ASAM for different initial weights.

for larger batch sizes, but SAM mitigates this effect compared to vanilla SGD. In addition, we observed that training sessions with vanilla SGD tend to have higher variance than training sessions with SAM, so we suggest that there may be further correlations between landscape sharpness, gradient variance, and training loss variance, which merit further investigation. A counter-intuitive observation is that low training loss may not always be a good sign in terms of generalization performance.

Lastly, experiment c showed that the removal of the batch normalization resulted in the dirstribution of weights significantly changing. The distribution of weights in the networks without batch normalization showed a much heavier tail, indicating that there is a large difference in the scale of the parameter. We have also seen that the ASAM algorithm generally performs better than the SAM algorithm on this task. Moreover, we saw a larger difference in the variance of the training loss in the SAM algorithm, while ASAM seemed to be much more constant for different weight initializations. Future work could expand on these experiments by testing the difference in performance of SAM and ASAM on more models and different data-sets.

REFERENCES

[1] Z. Xie, I. Sato, and M. Sugiyama, "A diffusion theory for deep learning dynamics: Stochastic gradient descent exponentially favors flat minima," type: article. [Online]. Available: http://arxiv.org/abs/2002.03495

[2] S. Jastrzebski, Z. Kenton, D. Arpit, N. Ballas, A. Fischer, Y. Bengio, and A. Storkey, "Finding flatter minima with SGD." [Online]. Available: https://openreview.net/forum?id=r1VF9dCUG

[3] P. Foret, A. Kleiner, H. Mobahi, and B. Neyshabur, "Sharpness-aware minimization for efficiently improving generalization." [Online]. Available: http://arxiv.org/abs/2010.01412

[4] J. Kwon, J. Kim, H. Park, and I. K. Choi, "ASAM: Adaptive sharpness-aware minimization for scale-invariant learning of deep neural networks," in *Proceedings of the 38th International Conference on Machine Learning*. PMLR, pp. 5905–5914, ISSN: 2640-3498. [Online]. Available: https://proceedings.mlr.press/v139/kwon21b.html

[5] M. Andriushchenko and N. Flammarion, "Understanding sharpness-aware minimization." [Online]. Available: https://openreview.net/forum?id=qXa0nhTRZGV

[6] "Towards generalization of medical imaging AI models: Sharpness-aware minimizers and beyond."

[7] C. Na, S. V. Mehta, and E. Strubell, "Train flat, then compress: Sharpness-aware minimization learns more compressible models," type: article. [Online]. Available: http://arxiv.org/abs/2205.12694

[8] D. Bahri, H. Mobahi, and Y. Tay, "Sharpness-aware minimization improves language model generalization," *arXiv preprint arXiv:2110.08529*, 2021.

[9] L. Dinh, R. Pascanu, S. Bengio, and Y. Bengio, "Sharp minima can generalize for deep nets," 2017. [Online]. Available: https://arxiv.org/abs/1703.04933

[10] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," 2015. [Online]. Available: https://arxiv.org/abs/1502.03167

[11] I. Loshchilov and F. Hutter, "Sgdr: Stochastic gradient descent with warm restarts," 2016. [Online]. Available: https://arxiv.org/abs/1608.03983
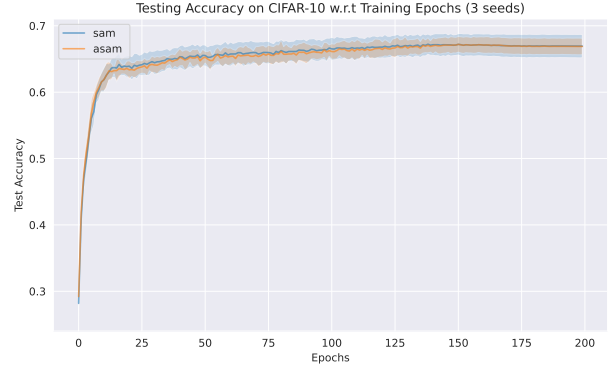
APPENDIX

Fig. 5. Testing accuracy on CIFAR-10 with SAM/ASAM for different initial weights.