OPT2023: 15th Annual Workshop on Optimization for Machine Learning

Revisiting Random Weight Perturbation for Efficiently Improving Generalization

Tao Li[1] LI.TAO@SJTU.EDU.CN
Weihao Yan[1]
Qinghua Tao[2]
Zehao Lei[1]
Yingwen Wu[1]
Kun Fang[1]
Mingzhen He[1]
Xiaolin Huang[1*] XIAOLINHUANG@SJTU.EDU.CN
[1]*Department of Automation, Shanghai Jiao Tong University*
[2]*ESAT-STADIUS, KU Leuven,* [*]*Corresponding author*

Abstract

Improving the generalization ability of modern deep neural networks (DNNs) is a fundamental problem in machine learning. Two branches of methods have been proposed to seek flat minima and improve generalization: one led by sharpness-aware minimization (SAM) minimizes the worst-case neighborhood loss through adversarial weight perturbation (AWP), and the other minimizes the expected Bayes objective with random weight perturbation (RWP). Although RWP has advantages in training time and is closely linked to AWP on a mathematical basis, its empirical performance always lags behind that of AWP. In this paper, we revisit RWP and analyze its convergence properties. We find that RWP requires a much larger perturbation magnitude than AWP, which leads to convergence issues. To resolve this, we propose m-RWP that incorporates the original loss objective to aid convergence, significantly lifting the performance of RWP. Compared with SAM, m-RWP is more efficient since it enables parallel computing of the two gradient steps and faster convergence, with comparable or even better performance[1].

1. Introduction

Modern deep neural networks (DNNs) are often over-parameterized and contain millions or even billions of parameters. As the number of parameters greatly exceeds that of samples, DNNs can easily memorize the entire training data and overfit them eventually, even with random labels [39]. Therefore, it is crucial to develop effective training algorithms that enable the network to achieve superior interpolation and generalize well beyond the training set [33].

Many works are devoted to improving the generalization ability of DNNs [9, 17, 35, 40, 41]. Following the idea that flat minima adapt better to the potential distribution shift between training and test data and thus exhibit better generalization [4, 15, 27], two prominent branches of methods have been proposed to seek such flat minima and show effective generalization improvement. The first formulates the optimization target as a min-max problem and tries to minimize the training loss under the worst-case adversarial weight perturbation (AWP), known as sharpness-aware minimization

1. The code is available at https://github.com/nblt/RWP.

(SAM) [9]. The second, represented by LPF-SGD [2], attempts to recover flat minima by minimizing the expected training loss under random weight perturbation (RWP). These two approaches are not only alike in formulation but can also be mathematically connected [32]. However, the empirical performance of RWP is commonly believed to be inferior to that of AWP [2, 29, 44], despite being computationally cheaper. The reason for this is that RWP is much weaker in perturbing the model than AWP, which can leverage the precise gradient information.

In this paper, we revisit RWP from a convergence perspective and aim to bridge the performance gap between these two types of perturbations. We show that RWP requires orders of perturbation magnitude larger than that of AWP for a similar perturbation strength, which can lead to convergence issues. To address this, we propose to incorporate the gradient of the original loss objective to improve convergence and guide the network towards better minima. We refer to our approach as ***m**(ixed)**-RWP***. Despite both SAM and m-RWP consuming two gradient steps for each iteration, the two in m-RWP are separable and thus can be efficiently computed *in parallel*, enabling the same training speed as regular SGD. In contrast, the two gradient steps in SAM are successive, resulting in a doubling of the training time. Another benefit of the two separable gradient steps in m-RWP is the ability to simultaneously use *two different batches* of data, further accelerating the convergence. In contrast, this is not permitted for SAM and can even have a detrimental effect on generalization performance. In summary, we make the following contributions:

- We analyze the convergence properties of SGD with RWP and propose to incorporate the gradient of the original loss objective to improve the convergence.
- We present m-RWP as an efficient alternative to SAM with comparable or even better performance. By parallelly computing the two gradient steps and utilizing two different data batches for each step, m-RWP halves the training time of SAM with faster convergence.
- We conduct extensive experiments with various architectures on benchmark image classification tasks to demonstrate the efficiency and effectiveness of our method.

2. Preliminary

Let $f(\boldsymbol{x}; \boldsymbol{w})$ be the neural network function with trainable parameters $\boldsymbol{w} \in \mathbb{R}^d$, where d is the number of parameters. The loss function over a pair of data point $(\boldsymbol{x}_i, \boldsymbol{y}_i)$ is denoted as $L(f(\boldsymbol{x}_i; \boldsymbol{w}), \boldsymbol{y}_i)$ (shorted for $L_i(\boldsymbol{w})$). Given the datasets $\mathcal{S} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^n$ drawn from data distribution $\mathcal{D}$ with i.i.d. condition, the empirical loss can be defined as $L(\boldsymbol{w}) = \frac{1}{n}\sum_{i=1}^n L_i(\boldsymbol{w})$.

Two branches of methods are proposed to pursue flat minima and better generalization ability. The first, known as sharpness-aware minimization [9], tries to minimize the worst-case loss in a neighborhood (defined by a norm ball) to bias training trajectories towards flat minima, i.e.,

$$L^{\text{SAM}}(\boldsymbol{w}) = \max_{\|\boldsymbol{\epsilon}_s\|_2 \leq \rho} L(\boldsymbol{w} + \boldsymbol{\epsilon}_s), \tag{1}$$

where ρ is the radius that controls the neighborhood size. Instead of posing the strict 'max-loss' over the neighborhood, the second, represented by LPF-SGD [2], adopts 'expected-loss' and minimizes the posterior (typically an isotropic Gaussian distribution) of the following Bayes objective [32]:

$$L^{\text{Bayes}}(\boldsymbol{w}) = \mathbb{E}_{\boldsymbol{\epsilon}_r \sim \mathcal{N}(0, \sigma^2 I)} L(\boldsymbol{w} + \boldsymbol{\epsilon}_r). \tag{2}$$

Such expected loss could effectively smooth the loss landscape and thus recover flat minima [2].

Adversarial Weight Perturbation (AWP). To optimize L^{SAM}, we first have to find the worst-case perturbations $\boldsymbol{\epsilon}_s^*$ for the max problem. Foret *et al.* [9] practically approximate (1) via the first-order expansion:

$$\boldsymbol{\epsilon}_s^* \approx \underset{\|\boldsymbol{\epsilon}_s\|_2 \leq \rho}{\arg\max}\, \boldsymbol{\epsilon}_s^\top \nabla_{\boldsymbol{w}} L(\boldsymbol{w}) = \rho \frac{\nabla_{\boldsymbol{w}} L(\boldsymbol{w})}{\|\nabla_{\boldsymbol{w}} L(\boldsymbol{w})\|_2}. \tag{3}$$

Then the gradient at the perturbed weight $\boldsymbol{w} + \boldsymbol{\epsilon_s^*}$ is computed for updating the model:

$$\nabla L^{\text{SAM}}(\boldsymbol{w}) \approx \nabla L(\boldsymbol{w})|_{\boldsymbol{w}+\boldsymbol{\epsilon}_s^*}. \tag{4}$$

Random Weight Perturbation (RWP). For optimizing L^{Bayes}, we similarly sample a random perturbation $\boldsymbol{\epsilon}_r$ and calculate the gradient at the perturbed weight $\boldsymbol{w} + \boldsymbol{\epsilon}_r$ for updating the model:

$$\nabla L^{\text{Bayes}}(\boldsymbol{w}) \approx \nabla L(\boldsymbol{w})|_{\boldsymbol{w}+\boldsymbol{\epsilon}_r}. \tag{5}$$

For modern DNNs, the loss function does not change with parameter scaling when ReLU-nonlinearities and batch normalization [16] are applied. Hence, it is essential to consider the filter-wise structure. Following the approach in [2], we practically generate the RWP from a filter-wise Gaussian distribution, i.e., $\boldsymbol{\epsilon}_r \sim \mathcal{N}\left(0, \sigma^2 diag(\|\boldsymbol{w}_1\|_2, \cdots, \|\boldsymbol{w}_k\|_2)\right)$, with σ controlling the variance.

RWP requires much larger magnitude. AWP is much more "effective" at perturbing the model compared to RWP. Consequently, in order to achieve a similar level of perturbation strength, the magnitude of RWP needs to be considerably larger than that of AWP. As dipicted in Figure A1, to attain a similar expected perturbed training loss $\mathbb{E}\left[L(\boldsymbol{w}^* + \boldsymbol{\epsilon})\right]$, the perturbation radius of RWP needs to be two orders of magnitude larger than that of AWP. Such large perturbations can introduce instability in training and cause convergence issues that degrade the performance.

3. Improving Random Weight Perturbation By Integrating Original Loss

We first investigate the convergence properties of the following RWP-SGD:

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \frac{\gamma_t}{|\mathcal{B}_t|} \sum_{i \in \mathcal{B}_t} \nabla L_i(\boldsymbol{w}_t + \boldsymbol{\epsilon}_r), \tag{6}$$

where $\mathcal{B}_t$ denotes the batch indices at time t. We make some standard assumptions on smoothness and bounded variance of stochastic gradients, which are typical as in [1, 8, 11, 18, 21].

Theorem 1 *(Convergence in convex settings) Let $\boldsymbol{\epsilon}_r \sim \mathcal{N}(0, \sigma^2 I_{d\times d})$ be the random perturbation and b be the batch size. Suppose that $L_i(\boldsymbol{w})$ is a convex function on $\mathbb{R}^d$ and $\boldsymbol{w}^*$ satisfies $\nabla L^{\text{Bayes}}(\boldsymbol{w}^*) = 0$. Consider the sequence $(\boldsymbol{w}_t)_{t\in\mathbb{N}}$ generated by (A1), with a stepsize satisfying $\gamma_t = \frac{\gamma}{\sqrt{t+1}}$ and $\gamma < \frac{1}{2\beta}$. Then we have*

$$\mathbb{E}\left[L(\bar{\boldsymbol{w}}^t) - \inf L\right] \leq \frac{\|\boldsymbol{w}_0 - \boldsymbol{w}^*\|^2}{2\gamma\sqrt{t}} + \frac{\gamma \log(t)}{\sqrt{t}} \left(\frac{\sigma_0^2}{b} + \sigma_L^*\right) + \alpha\sigma\sqrt{d}, \tag{7}$$

where $\bar{\boldsymbol{w}} = \sum_{k=0}^{t-1} p_{t,k} \boldsymbol{w}_k$, with $p_{t,k} = \frac{\gamma_k(1-2\gamma_k\beta)}{\sum_{k=0}^{t-1}\gamma_k(1-2\gamma_k\beta)}$ and $\sigma_L^* = \mathbb{E}\left[\|\nabla L(\boldsymbol{w}^* + \boldsymbol{\epsilon}_r)\|^2\right]$. Furthermore, if $L(\boldsymbol{w})$ is strongly convex, i.e., $\nabla^2 L(\boldsymbol{w}) \geq mI$, we have $\sigma_L^* \geq m^2\sigma^2 d$.

We present full assumptions and proof in Appendix A1 and make the following remarks: (1) The first two terms decrease at a rate of $\mathcal{O}(\frac{1}{\sqrt{t}})$ and $\mathcal{O}(\frac{\log(t)}{\sqrt{t}})$, respectively, which is consistent with the convergence rate of regular SGD. (2) The third term is a positive constant proportional to σ, which prevents the effective reduction of the loss at the end of training. Intuitively, the 'smoothed' loss function $L^{\text{Bayes}}(\boldsymbol{w})$ provides an upper bound on the original loss function $L(\boldsymbol{w})$ for convex functions, and their respective minima would be different, leading to the minima gap. (3) σ_L^* is the variance term introduced by random weight perturbation. It can remain large when $\|\boldsymbol{\epsilon}_r\|_2$ is large, thereby slowing down the convergence.

Improving convergence by integrating original loss. To improve the convergence of RWP-SGD, we propose to combine the original loss with the expected Bayes loss, i.e.,

$$L^{\text{m}}(\boldsymbol{w}) = \lambda L^{\text{Bayes}}(\boldsymbol{w}) + (1-\lambda)L(\boldsymbol{w}), \tag{8}$$

where λ is a pre-given balance coefficient. These two loss terms are complementary to each other: the first $L^{\text{Bayes}}(\boldsymbol{w})$ provides a smoothed landscape that biases the network towards flat region, while the second $L(\boldsymbol{w})$ helps recover the necessary local information and better locates the minima that contributes to high performance. These two together could provide a both 'local' and 'global' viewing of the landscape — by optimizing $L^{\text{m}}(\boldsymbol{w})$, a good solution can be expected.

Efficient Parallel Training. Both SAM and m-RWP involve two gradient steps for each iteration, i.e., $\nabla L(\boldsymbol{w})$ and $\nabla L(\boldsymbol{w}+\boldsymbol{\epsilon})$. The two steps in m-RWP are separable, whereas, in SAM, they are computed sequentially. This allows us to half the training time of m-RWP by parallel computing.

Different Data Batches Accelerate Convergence. For m-RWP, we can use *two different data batches* to compute the two gradient steps. This virtually enlarges the batch size and reduces the gradient variance term in (7), which is $\frac{\sigma_0^2}{b}$, by a factor of $2\lambda^2 - 2\lambda + 1$. This, in turn, enhances the convergence rate. However, we find that such modifications undesirably destroy the generalization performance of SAM to that of SGD, as shown in Table A1. We present the convergence analysis of m-RWP in Appendix A1.2 and the training curves comparison of different methods in Appendix A6.

4. Experiments

In this section, we conduct extensive experiments to demonstrate the efficiency and effectiveness of our proposed m-RWP algorithm. We begin by introducing the experimental setup and then evaluate the performance over standard benchmark datasets.

Setup. We experiment over three benchmark image classification tasks, including CIFAR-10, CIFAR-100 [25], and ImageNet [3], and evaluate across various representative DNN architectures, including VGG [34], ResNet [13], WideResNet [38], and ViT [5]. We compare four training schemes, including SGD, SAM, RWP and m-RWP, and place the detailed settings in Appendix A5.

Results. We first focus on the CIFAR-10 and CIFAR-100 datasets. We compare the final test accuracy, total computation (in FLOPs), and training time for different schemes. Detailed comparisons are presented in Table 1. It is worth noting that while both SAM and m-RWP require the same amount of computation, which is twice that of SGD, the training time of m-RWP can be reduced by half compared to SAM through parallel computing. RWP does not bring consistent improvement over SGD, showing its sensitivity to different architectures, and has a significant performance gap compared to SAM, especially on larger models. m-RWP consistently outperforms RWP, enhancing

Table 1: Results on CIFAR-10/100. We set the computation (FLOPs) and training time of SGD as $1\times$. The best accuracy is in bold and the second best is underlined.

Model	Method	CIFAR-10	CIFAR-100	FLOPs	Time
VGG16-BN	SGD	94.96±0.15	75.43±0.29	1×	1×
	SAM	<u>95.43</u>±0.11	<u>76.74</u>±0.22	2×	2×
	RWP	94.97±0.01	76.31±0.15	1×	1×
	m-RWP	**95.61**±0.23	**77.85**±0.17	2×	1×
ResNet-18	SGD	96.10±0.08	78.10±0.39	1×	1×
	SAM	<u>96.50</u>±0.08	<u>80.22</u>±0.23	2×	2×
	RWP	96.17±0.31	80.08±0.11	1×	1×
	m-RWP	**96.68**±0.17	**81.25**±0.13	2×	1×
WRN-28-10	SGD	96.85±0.05	82.51±0.24	1×	1×
	SAM	**97.37**±0.02	**84.44**±0.03	2×	2×
	RWP	96.73±0.12	83.40±0.08	1×	1×
	m-RWP	<u>97.28</u>±0.09	<u>84.37</u>±0.12	2×	1×
ViT-S	Adam	86.60±0.03	63.66±0.28	1×	1×
	SAM	<u>87.48</u>±0.28	<u>64.83</u>±0.24	2×	2×
	RWP	86.53±0.04	63.07±0.41	1×	1×
	m-RWP	**88.18**±0.19	**66.13**±0.03	2×	1×

performance by 1.6% on CIFAR-10 and 3.1% on CIFAR-100, thereby confirming its effectiveness in improving generalization. Besides, m-RWP achieves very competitive performance against SAM: for example, it outperforms SAM significantly by 1.1% with VGG16-BN and 1.0% with ResNet-18 on CIFAR-100. On the larger WideResNet models, m-RWP performs comparably to SAM, e.g., +0.09% with WideResNet-16-8 and -0.07% with WideResNet-28-10 on CIFAR-100.

Next, we evaluate our proposed scheme on the ImageNet dataset, which has a substantially larger scale than CIFAR datasets. In the case of SAM, the $2\times$ longer training time would be prohibitively slow, making it essential to improve the training efficiency. We evaluate over ResNet-18 and ResNet-50, and present the results in Table 2. We observe that m-RWP significantly outperforms SAM, achieving 71.58% accuracy (+0.81%) with ResNet-18, and 78.04% accuracy (+0.89%) with ResNet-50. We note that models trained on ImageNet are typically under-trained and thus the faster convergence of m-RWP offers even more significant advantages over SAM.

Table 2: Results on ImageNet. We set the computation (FLOPs) and training time of SGD as $1\times$.

Model	Training	Top-1 Accuracy	Top-5 Accuracy	FLOPs	Time
ResNet-18	SGD	70.46	89.79	1×	1×
	SAM	<u>70.77</u>	<u>89.83</u>	2×	2×
	RWP	70.65	89.60	1×	1×
	m-RWP	**71.58**	**90.31**	2×	1×
ResNet-50	SGD	76.83	93.55	1×	1×
	SAM	<u>77.15</u>	<u>93.55</u>	2×	2×
	RWP	76.32	92.99	1×	1×
	m-RWP	**78.04**	**93.91**	2×	1×

5. Conclusion

In this work, we revisit the use of random weight perturbation for improving generalization performance. We demonstrate that random weight perturbation requires a much larger perturbation magnitude than the adversarial one, which leads to convergence issues. To address this problem, we propose m-RWP that incorporates the original loss objective to aid convergence and significantly enhance the performance. Compared with the current state-of-the-art SAM that adopts adversarial weight perturbation, our extensive experiments show that m-RWP achieves comparable or even better performance. Moreover, m-RWP offers greater efficiency by enabling parallel computing of the two gradient steps and faster convergence by utilizing different data batches for each step.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (No. 62376155, 61977046), Shanghai Municipal Science and Technology Major Project (No. 2021SHZDZX0102) and the Shanghai Science and Technology Program under Grant (No. 21JC1400600).

References

[1] Maksym Andriushchenko and Nicolas Flammarion. Towards understanding sharpness-aware minimization. In *International Conference on Machine Learning (ICML)*, 2022.

[2] Devansh Bisla, Jing Wang, and Anna Choromanska. Low-pass filtering sgd for recovering flat optima in the deep learning optimization landscape. In *International Conference on Artificial Intelligence and Statistics (AISTATIS)*, 2022.

[3] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[4] Laurent Dinh, Razvan Pascanu, Samy Bengio, and Yoshua Bengio. Sharp minima can generalize for deep nets. In *International Conference on Machine Learning (ICML)*, 2017.

[5] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations (ICLR)*, 2021.

[6] Jiawei Du, Hanshu Yan, Jiashi Feng, Joey Tianyi Zhou, Liangli Zhen, Rick Siow Mong Goh, and Vincent YF Tan. Efficient sharpness-aware minimization for improved training of neural networks. In *International Conference on Learning Representations (ICLR)*, 2022.

[7] Jiawei Du, Daquan Zhou, Jiashi Feng, Vincent YF Tan, and Joey Tianyi Zhou. Sharpness-aware training for free. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

[8] John C Duchi, Peter L Bartlett, and Martin J Wainwright. Randomized smoothing for stochastic optimization. *SIAM Journal on Optimization*, 22(2):674–701, 2012.

[9] Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. In *International Conference on Learning Representations (ICLR)*, 2020.

[10] Guillaume Garrigos and Robert M Gower. Handbook of convergence theorems for (stochastic) gradient methods. *arXiv preprint arXiv:2301.11235*, 2023.

[11] Saeed Ghadimi and Guanghui Lan. Stochastic first-and zeroth-order methods for nonconvex stochastic programming. *SIAM Journal on Optimization*, 23(4):2341–2368, 2013.

[12] Behrooz Ghorbani, Shankar Krishnan, and Ying Xiao. An investigation into neural net optimization via hessian eigenvalue density. In *International Conference on Machine Learning*, 2019.

[13] 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 (CVPR)*, 2016.

[14] Sepp Hochreiter and Jürgen Schmidhuber. Simplifying neural nets by discovering flat minima. In *Advances in Neural Information Processing Systems (NeurIPS)*, 1994.

[15] Sepp Hochreiter and Jürgen Schmidhuber. Flat minima. *Neural computation*, 1997.

[16] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning (ICML)*, 2015.

[17] Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. Averaging weights leads to wider optima and better generalization. *arXiv preprint arXiv:1803.05407*, 2018.

[18] Weisen Jiang, Hansi Yang, Yu Zhang, and James Kwok. An adaptive policy to employ sharpness-aware minimization. In *International Conference on Learning Representations (ICLR)*, 2023.

[19] Yiding Jiang*, Behnam Neyshabur*, Hossein Mobahi, Dilip Krishnan, and Samy Bengio. Fantastic generalization measures and where to find them. In *International Conference on Learning Representations (ICLR)*, 2020.

[20] Chi Jin, Praneeth Netrapalli, Rong Ge, Sham M Kakade, and Michael I Jordan. On nonconvex optimization for machine learning: Gradients, stochasticity, and saddle points. *Journal of the ACM (JACM)*, 2021.

[21] Hamed Karimi, Julie Nutini, and Mark Schmidt. Linear convergence of gradient and proximal-gradient methods under the polyak-łojasiewicz condition. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 2016.

[22] Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. In *International Conference on Learning Representations (ICLR)*, 2017.

[23] Minyoung Kim, Da Li, Shell X Hu, and Timothy Hospedales. Fisher sam: Information geometry and sharpness aware minimisation. In *International Conference on Machine Learning (ICML)*, 2022.

[24] Diederik P. Kingma and Jimmy Lei Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015.

[25] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. *Technical Report*, 2009.

[26] Jungmin Kwon, Jeongseop Kim, Hyunseo Park, and In Kwon Choi. Asam: Adaptive sharpness-aware minimization for scale-invariant learning of deep neural networks. In *International Conference on Machine Learning (ICML)*, 2021.

[27] Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the loss landscape of neural nets. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

[28] Yong Liu, Siqi Mai, Xiangning Chen, Cho-Jui Hsieh, and Yang You. Towards efficient and scalable sharpness-aware minimization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (CVPR)*, 2022.

[29] Yong Liu, Siqi Mai, Minhao Cheng, Xiangning Chen, Cho-Jui Hsieh, and Yang You. Random sharpness-aware minimization. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

[30] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016.

[31] Peng Mi, Li Shen, Tianhe Ren, Yiyi Zhou, Xiaoshuai Sun, Rongrong Ji, and Dacheng Tao. Make sharpness-aware minimization stronger: A sparsified perturbation approach. *arXiv preprint arXiv:2210.05177*, 2022.

[32] Thomas Möllenhoff and Mohammad Emtiyaz Khan. SAM as an optimal relaxation of Bayes. In *International Conference on Learning Representations (ICLR)*, 2023. URL https://openreview.net/forum?id=k4fevFqSQcX.

[33] Behnam Neyshabur, Srinadh Bhojanapalli, David McAllester, and Nati Srebro. Exploring generalization in deep learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

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

[35] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[36] Ziqiao Wang and Yongyi Mao. On the generalization of models trained with sgd: Information-theoretic bounds and implications. In *International Conference on Learning Representations*, 2021.

[37] Wei Wen, Yandan Wang, Feng Yan, Cong Xu, Chunpeng Wu, Yiran Chen, and Hai Li. Smoothout: Smoothing out sharp minima to improve generalization in deep learning. *arXiv preprint arXiv:1805.07898*, 2018.

[38] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In Richard C. Wilson, Edwin R. Hancock, and William A. P. Smith, editors, *British Machine Vision Conference (BMVC)*, 2016.

[39] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning (still) requires rethinking generalization. *Communications of the ACM*, 64(3): 107–115, 2021.

[40] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations (ICLR)*, 2018.

[41] Michael Zhang, James Lucas, Jimmy Ba, and Geoffrey E Hinton. Lookahead optimizer: k steps forward, 1 step back. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[42] Yang Zhao, Hao Zhang, and Xiuyuan Hu. Penalizing gradient norm for efficiently improving generalization in deep learning. In *International Conference on Machine Learning (ICML)*, 2022.

[43] Yang Zhao, Hao Zhang, and Xiuyuan Hu. SS-SAM: Stochastic scheduled sharpness-aware minimization for efficiently training deep neural networks. *arXiv preprint arXiv:2203.09962*, 2022.

[44] Yaowei Zheng, Richong Zhang, and Yongyi Mao. Regularizing neural networks via adversarial model perturbation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[45] Mo Zhou, Tianyi Liu, Yan Li, Dachao Lin, Enlu Zhou, and Tuo Zhao. Toward understanding the importance of noise in training neural networks. In *International Conference on Machine Learning (ICML)*, 2019.

[46] Juntang Zhuang, Boqing Gong, Liangzhe Yuan, Yin Cui, Hartwig Adam, Nicha Dvornek, Sekhar Tatikonda, James Duncan, and Ting Liu. Surrogate gap minimization improves sharpness-aware training. In *International Conference on Learning Representations (ICLR)*, 2022.

Appendix A1. Missing Proofs

A1.1. Proof of Theorem 4.1

In this part, we prove Theorem 4.1 in the main paper, which provides analytical results on the convergence of the following RWP-SGD algorithm:

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \frac{\gamma_t}{|\mathcal{B}_t|}\sum_{i\in\mathcal{B}_t}\nabla L_i(\boldsymbol{w}_t + \boldsymbol{\epsilon}_r). \tag{A1}$$

The proof is based on [2, 8, 10]. Before proceeding with the proof, we need the following necessary lemmas and assumptions.

Lemma 2 *(Bisla et al. [2]) Let $L(\boldsymbol{w}) : \mathbb{R}^d \to \mathbb{R}$ be α-Lipschitz continuous and β-smooth w.r.t. l_2-norm. The smoothed loss function is defined as $L^{\text{Bayes}}(\boldsymbol{w}) \triangleq \mathbb{E}_{\boldsymbol{\epsilon}_r\sim\mathcal{N}(0,\sigma^2 I)}L(\boldsymbol{w}+\boldsymbol{\epsilon}_r)$. Then the following properties hold:*

1. *$L^{\text{Bayes}}(\boldsymbol{w})$ is α-Lipschitz continuous.*
2. *$L^{\text{Bayes}}(\boldsymbol{w})$ is continuously differentiable; moreover, its gradient is $\min\{\frac{\alpha}{\sigma}, \beta\}$-Lipschitz continuous, i.e., $L^{\text{Bayes}}(\boldsymbol{w})$ is $\min\{\frac{\alpha}{\sigma}, \beta\}$-smooth.*
3. *If $L(\boldsymbol{w})$ is convex, $L(\boldsymbol{w}) \le L^{\text{Bayes}}(\boldsymbol{w}) \le L(\boldsymbol{w}) + \alpha\sigma\sqrt{d}$.*

Lemma 3 *(Garrigos et al. [10]) If $L(\boldsymbol{w})$ is convex and β-smooth, for all $\boldsymbol{w}_1, \boldsymbol{w}_2 \in \mathbb{R}^d$, we have*

$$\frac{1}{2\beta}\|\nabla L(\boldsymbol{w}_2) - \nabla L(\boldsymbol{w}_1)\|^2 \le L(\boldsymbol{w}_2) - L(\boldsymbol{w}_1) - \langle\nabla L(\boldsymbol{w}_1), \boldsymbol{w}_2 - \boldsymbol{w}_1\rangle.$$

Assumption 1 (Bounded variance)**.** $\mathbb{E}\left[\|\nabla L_i(\boldsymbol{w}) - \nabla L(\boldsymbol{w})\|^2\right] \le \sigma_0^2$ for all $\boldsymbol{w} \in \mathbb{R}^d$ and $i \in [n]$.
Assumption 2 (Individual α-Lipschitz continuous and β-smoothness)**.** There exist $\alpha, \beta \ge 0$ such that $\|L_i(\boldsymbol{w}) - L_i(\boldsymbol{v})\| \le \alpha\|\boldsymbol{w} - \boldsymbol{v}\|$, $\|\nabla L_i(\boldsymbol{w}) - \nabla L_i(\boldsymbol{v})\| \le \beta\|\boldsymbol{w} - \boldsymbol{v}\|$ for all $\boldsymbol{w}, \boldsymbol{v} \in \mathbb{R}^d$, $i \in [n]$.

Theorem 4.1 *Let $\boldsymbol{\epsilon}_r \sim \mathcal{N}(0, \sigma^2 I_{d\times d})$ be the random perturbation and b be the batch size. Suppose that $L_i(\boldsymbol{w})$ is a convex function on $\mathbb{R}^d$ and $\boldsymbol{w}^*$ satisfies $\nabla L^{\text{Bayes}}(\boldsymbol{w}^*) = 0$. Consider the sequence $(\boldsymbol{w}_t)_{t\in\mathbb{N}}$ generated by (A1), with a stepsize satisfying $\gamma_t = \frac{\gamma}{\sqrt{t+1}}$ and $\gamma < \frac{1}{2\beta}$. Then we have*

$$\mathbb{E}\left[L(\bar{\boldsymbol{w}}^t) - \inf L\right] \le \frac{\|\boldsymbol{w}_0 - \boldsymbol{w}^*\|^2}{2\gamma\sqrt{t}} + \frac{\gamma\log(t)}{\sqrt{t}}\left(\frac{\sigma_0^2}{b} + \sigma_L^*\right) + \alpha\sigma\sqrt{d},$$

where $\bar{\boldsymbol{w}} = \sum_{k=0}^{t-1} p_{t,k}\boldsymbol{w}_k$, with $p_{t,k} = \frac{\gamma_k(1-2\gamma_k\beta)}{\sum_{k=0}^{t-1}\gamma_k(1-2\gamma_k\beta)}$ and $\sigma_L^* = \mathbb{E}\left[\|\nabla L(\boldsymbol{w}^* + \boldsymbol{\epsilon}_r)\|^2\right]$. Furthermore, if $L(\boldsymbol{w})$ is strongly convex, i.e., $\nabla^2 L(\boldsymbol{w}) \ge mI$, we have $\sigma_L^* \ge m^2\sigma^2 d$.

Proof Let $L_{\mathcal{B}}(\boldsymbol{w}) \overset{\text{def}}{=} \frac{1}{|\mathcal{B}|}\sum_{i\in\mathcal{B}}\nabla L_i(\boldsymbol{w})$. We first derive the following inequality that holds true:

$$\|\nabla L_{\mathcal{B}_t}(\boldsymbol{w} + \boldsymbol{\epsilon}_r)\|^2 \le 2\|\nabla L_{\mathcal{B}_t}(\boldsymbol{w} + \boldsymbol{\epsilon}_r) - \nabla L_{\mathcal{B}_t}(\boldsymbol{w}^* + \boldsymbol{\epsilon}_r)\|^2 + 2\|\nabla L_{\mathcal{B}_t}(\boldsymbol{w}^* + \boldsymbol{\epsilon}_r)\|^2.$$

Applying Lemma 3 and taking the expectation over both sides, we obtain:

$$\mathbb{E}_{\mathcal{B}_t,\boldsymbol{\epsilon}_r}\left[\|\nabla L_{\mathcal{B}_t}(\boldsymbol{w} + \boldsymbol{\epsilon}_r)\|^2\right] \le 4\beta(L^{\text{Bayes}}(\boldsymbol{w}) - L^{\text{Bayes}}(\boldsymbol{w}^*)) + \frac{2\sigma_0^2}{b} + 2\sigma_L^*,$$

where the last two terms follow from

$$\begin{aligned}
&\mathbb{E}_{\mathcal{B}_t,\boldsymbol{\epsilon}_r}\left[\|\nabla L_{\mathcal{B}_t}(\boldsymbol{w}^*+\boldsymbol{\epsilon}_r)\|^2\right]\\
&=\mathbb{E}_{\mathcal{B}_t,\boldsymbol{\epsilon}_r}\left[\|\nabla L_{\mathcal{B}_t}(\boldsymbol{w}^*+\boldsymbol{\epsilon}_r)-\nabla L(\boldsymbol{w}^*+\boldsymbol{\epsilon}_r)+\nabla L(\boldsymbol{w}^*+\boldsymbol{\epsilon}_r)\|^2\right]\\
&=\mathbb{E}_{\mathcal{B}_t,\boldsymbol{\epsilon}_r}\left[\|\nabla L_{\mathcal{B}_t}(\boldsymbol{w}^*+\boldsymbol{\epsilon}_r)-\nabla L(\boldsymbol{w}^*+\boldsymbol{\epsilon}_r)\|^2\right]+\mathbb{E}_{\boldsymbol{\epsilon}_r}\left[\|\nabla L(\boldsymbol{w}^*+\boldsymbol{\epsilon}_r)\|^2\right]\\
&\leq \frac{\sigma_0^2}{b}+\sigma_L^*.
\end{aligned}$$

Next, we analyze the behaviour of $\|\boldsymbol{w}_{t+1}-\boldsymbol{w}^*\|^2$. By developing squares, we obtain:

$$\|\boldsymbol{w}_{t+1}-\boldsymbol{w}^*\|^2=\|\boldsymbol{w}_t-\boldsymbol{w}^*\|^2-2\gamma_t\langle\nabla L_{\mathcal{B}_t}(\boldsymbol{w}_t+\boldsymbol{\epsilon}_r),\boldsymbol{w}_t-\boldsymbol{w}^*\rangle+\gamma_t^2\|\nabla L_{\mathcal{B}_t}(\boldsymbol{w}_t+\boldsymbol{\epsilon}_r)\|^2.$$

Taking expectation conditioned on $\boldsymbol{w}_t$ and utilizing the convexity of $L^{\text{Bayes}}(\boldsymbol{w})$, we obtain:

$$\begin{aligned}
&\mathbb{E}_t\left[\|\boldsymbol{w}_{t+1}-\boldsymbol{w}^*\|^2\right]\\
&=\|\boldsymbol{w}_t-\boldsymbol{w}^*\|^2-2\gamma_t\langle\nabla L^{\text{Bayes}}(\boldsymbol{w}_t),\boldsymbol{w}_t-\boldsymbol{w}^*\rangle+\gamma_t^2\mathbb{E}\left[\|\nabla L((\boldsymbol{w}_t+\boldsymbol{\epsilon}_r)\|^2\right]\\
&\leq\|\boldsymbol{w}_t-\boldsymbol{w}^*\|^2+2\gamma_t\left(2\gamma_t\beta-1\right)\left(L^{\text{Bayes}}(\boldsymbol{w}_t)-L^{\text{Bayes}}(\boldsymbol{w}^*)\right)+2\gamma_t^2(\frac{\sigma_0^2}{b}+\sigma_L^*).
\end{aligned}$$

Further taking expectation and summing over $k=0,\cdots,t-1$, we have:

$$\begin{aligned}
&2\sum_{k=0}^{t-1}\gamma_k\left(1-2\gamma_k\beta\right)\mathbb{E}\left[L^{\text{Bayes}}(\boldsymbol{w}_k)-L^{\text{Bayes}}(\boldsymbol{w}^*)\right]\\
&\leq\|\boldsymbol{w}_0-\boldsymbol{w}^*\|^2-\mathbb{E}\left[\|\boldsymbol{w}_t-\boldsymbol{w}^*\|^2\right]+2(\frac{\sigma_0^2}{b}+\sigma_L^*)\sum_{k=0}^{t-1}\gamma_k^2.
\end{aligned}$$

Dividing both sides by $2\sum_{k=0}^{t-1}\gamma_k\left(1-2\gamma_k\beta\right)$, we obtain:

$$\begin{aligned}
&\sum_{k=0}^{t-1}\mathbb{E}\left[\frac{\gamma_k\left(1-2\gamma_k\beta\right)}{\sum_{k=0}^{t-1}\gamma_k\left(1-2\gamma_k\beta\right)}\left[L^{\text{Bayes}}(\boldsymbol{w}_k)-L^{\text{Bayes}}(\boldsymbol{w}^*)\right]\right]\\
&\leq\frac{\|\boldsymbol{w}_0-\boldsymbol{w}^*\|^2}{2\sum_{k=0}^{t-1}\gamma_k\left(1-2\gamma_k\beta\right)}+\frac{(\frac{\sigma_0^2}{b}+\sigma_L^*)\sum_{k=0}^{t-1}\gamma_k^2}{\sum_{k=0}^{t-1}\gamma_k\left(1-2\gamma_k\beta\right)}.
\end{aligned}$$

Define $p_{t,k}\overset{\text{def}}{=}\frac{\gamma_k(1-2\gamma_kL_1)}{\sum_{k=0}^{t-1}\gamma_k(1-2\gamma_kL_1)}$, we observe $p_{t,k}\geq 0$ and $\sum_{k=0}^{t-1}p_{t,k}=1$. Using that $L^{\text{Bayes}}(\boldsymbol{w})$ is convex together with Jensen's inequality gives:

$$\begin{aligned}
\mathbb{E}\left[L^{\text{Bayes}}(\bar{\boldsymbol{w}}^t)-L^{\text{Bayes}}(\boldsymbol{w}^*)\right]&\leq\sum_{k=0}^{t-1}\mathbb{E}\left[\frac{\gamma_k\left(1-2\gamma_k\beta\right)}{\sum_{k=0}^{t-1}\gamma_k\left(1-2\gamma_k\beta\right)}\left[L^{\text{Bayes}}(\boldsymbol{w}_k)-L^{\text{Bayes}}(\boldsymbol{w}^*)\right]\right]\\
&\leq\frac{\|\boldsymbol{w}_0-\boldsymbol{w}^*\|^2}{2\sum_{k=0}^{t-1}\gamma_k\left(1-2\gamma_k\beta\right)}+\frac{(\frac{\sigma_0^2}{b}+\sigma_L^*)\sum_{k=0}^{t-1}\gamma_k^2}{\sum_{k=0}^{t-1}\gamma_k\left(1-2\gamma_k\beta\right)}.
\end{aligned}\tag{A2}$$

Finally, with the integral bound

$$\sum_{k=0}^{t-1}\gamma_k^2=\gamma^2\sum_{k=0}^{t-1}\frac{1}{k+1}\leq\gamma^2\int_{k=0}^{t-1}\frac{1}{k+1}\mathrm{d}k=\gamma^2\log(t),$$
$$\sum_{k=0}^{t-1}\gamma_k\geq\int_{k=1}^{t-1}\frac{\gamma}{\sqrt{k+1}}\mathrm{d}k=2\gamma\left(\sqrt{t}-\sqrt{2}\right),$$

we have the following equality for large enough t:

$$\begin{aligned}\sum_{k=0}^{t-1}\gamma_k\left(1-2\gamma_k\beta\right)&\geq 2\gamma\left(\sqrt{t}-\sqrt{2}-\gamma\beta\log(t)\right)\\&\geq 2\gamma\left(\sqrt{t}-\sqrt{2}-\log(\sqrt{k})\right)\\&\geq\gamma\sqrt{t}.\end{aligned}$$

Applying the above equality into (A2) and with 2, we arrive at the result:

$$\begin{aligned}\mathbb{E}\left[L(\bar{\boldsymbol{w}}^t)-L(\boldsymbol{w}^*)\right]&\leq\frac{\|\boldsymbol{w}_0-\boldsymbol{w}^*\|^2}{2\sum_{k=0}^{t-1}\gamma_k\left(1-2\gamma_k\beta\right)}+\frac{(\frac{\sigma_0^2}{b}+\sigma_L^*)\sum_{k=0}^{t-1}\gamma_k^2}{\sum_{k=0}^{t-1}\gamma_k\left(1-2\gamma_k\beta\right)}+\alpha\sigma\sqrt{d}\\&\leq\frac{\|\boldsymbol{w}_0-\boldsymbol{w}^*\|^2}{2\gamma\sqrt{t}}+\frac{(\frac{\sigma_0^2}{b}+\sigma_L^*)\gamma^2\log(t)}{\gamma\sqrt{t}}+\alpha\sigma\sqrt{d}\\&=\frac{\|\boldsymbol{w}_0-\boldsymbol{w}^*\|^2}{2\gamma\sqrt{t}}+\frac{\gamma\log(t)}{\sqrt{t}}(\frac{\sigma_0^2}{b}+\sigma_L^*)+\alpha\sigma\sqrt{d}.\end{aligned}$$

Furthermore, if $\nabla^2 L(\boldsymbol{w})\geq mI$, let $\boldsymbol{w}_L^*\overset{\text{def}}{=}\arg\min_{\boldsymbol{w}}L(\boldsymbol{w})$, then we have the following bound:

$$\mathbb{E}\left[\|\nabla L(\boldsymbol{w}^*+\boldsymbol{\epsilon}_r)\|^2\right]\geq\mathbb{E}\left[\|\nabla L(\boldsymbol{w}_L^*+\boldsymbol{\epsilon}_r)\|^2\right]\geq m^2\mathbb{E}\left[\|\boldsymbol{\epsilon}_r\|^2\right]=m^2\sigma^2 d.$$

■

A1.2. Convergence analysis of m-RWP

In the following, we provide the convergence analysis of the m-RWP algorithm, which integrates the gradient of the original loss objective into RWP-SGD to improve the convergence.

Theorem A2 *Let $\boldsymbol{\epsilon}_r\sim\mathcal{N}(0,\sigma^2 I_{d\times d})$ be the random perturbation and b be the batch size. Suppose $L_i(\boldsymbol{w})$ is a convex function on $\mathbb{R}^d$ and $\boldsymbol{w}^*$ satisfies $\nabla L^{\mathrm{m}}(\boldsymbol{w}^*)=0$. Consider the sequence $(\boldsymbol{w}_t)_{t\in\mathbb{N}}$ generated by Algorithm 2, with a stepsize satisfying $\gamma_t=\frac{\gamma}{\sqrt{t+1}}$ and $\gamma<\frac{1}{2\beta}$. Then we have:*

$$\mathbb{E}\left[L(\bar{\boldsymbol{w}}^t)-\inf L\right]\leq\frac{\|\boldsymbol{w}_0-\boldsymbol{w}^*\|^2}{2\gamma\sqrt{t}}+\frac{\gamma\log(t)}{\sqrt{t}}\left(\frac{2\lambda^2-2\lambda+1}{b}\sigma_0^2+\lambda^2\sigma_m^*\right)+\alpha\sigma\lambda\sqrt{d},$$

where $\bar{\boldsymbol{w}}=\sum_{k=0}^{t-1}p_{t,k}\boldsymbol{w}_k$ with $p_{t,k}=\frac{\gamma_k(1-2\gamma_k\beta)}{\sum_{k=0}^{t-1}\gamma_k(1-2\gamma_k\beta)}$ and $\sigma_m^*=\mathbb{E}\left[\|\nabla L(\boldsymbol{w}^*+\boldsymbol{\epsilon}_r)-\nabla L^{\mathrm{Bayes}}(\boldsymbol{w}^*)\|^2\right]$.

Proof The schematic derivation of this proof is similar to that of Theorem 4.1. Firstly, we recall the update rule in m-RWP:

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \gamma_t \left[\frac{\lambda}{|\mathcal{B}_{t,1}|} \sum_{i \in \mathcal{B}_{t,1}} \nabla L_i(\boldsymbol{w}_t + \boldsymbol{\epsilon}_r) + \frac{1-\lambda}{|\mathcal{B}_{t,2}|} \sum_{i \in \mathcal{B}_{t,2}} \nabla L_i(\boldsymbol{w}_t) \right].$$

Let $L_{\mathcal{B}}(\boldsymbol{w}) \overset{\text{def}}{=} \frac{1}{|\mathcal{B}|} \sum_{i \in \mathcal{B}} \nabla L_i(\boldsymbol{w})$ and $\nabla L^m_{\mathcal{B}}(\boldsymbol{w}) \overset{\text{def}}{=} \lambda \nabla L_{\mathcal{B}_{t,1}}(\boldsymbol{w} + \boldsymbol{\epsilon}_r) + (1-\lambda) \nabla L_{\mathcal{B}_{t,2}}(\boldsymbol{w})$, we first have the following equality:

$$\|\nabla L^m_{\mathcal{B}}(\boldsymbol{w})\|^2 \leq 2\|\nabla L^m_{\mathcal{B}}(\boldsymbol{w}) - \nabla L^m_{\mathcal{B}}(\boldsymbol{w}^*)\|^2 + 2\|\nabla L^m_{\mathcal{B}}(\boldsymbol{w}^*)\|^2.$$

Applying Lemma 3 and taking expectation over both sides, we obtain:

$$\mathbb{E}_{\mathcal{B}_{t,1}, \mathcal{B}_{t,2}, \boldsymbol{\epsilon}_r} \left[\|\nabla L^m_{\mathcal{B}}(\boldsymbol{w})] \right] \leq 4\beta (L^{\mathrm{m}}(\boldsymbol{w}) - L^{\mathrm{m}}(\boldsymbol{w}^*)) + \frac{2\sigma_0^2(2\lambda^2 - 2\lambda + 1)}{b} + 2\lambda^2 \sigma_m^*.$$

where the last two terms follow from that

$$\begin{aligned}
&\mathbb{E}_{\mathcal{B}_{t,1}, \mathcal{B}_{t,2}, \boldsymbol{\epsilon}_r} \left[\|\nabla L^m_{\mathcal{B}}(\boldsymbol{w}^*)\|^2 \right] \\
&= \mathbb{E}_{\mathcal{B}_{t,1}, \mathcal{B}_{t,2}, \boldsymbol{\epsilon}_r} \left[\|\nabla L^m_{\mathcal{B}}(\boldsymbol{w}^*) - \mathbb{E}_{\mathcal{B}_{t,1}, \mathcal{B}_{t,2}} \left[\nabla L^m_{\mathcal{B}}(\boldsymbol{w}^*) \right] + \mathbb{E}_{\mathcal{B}_{t,1}, \mathcal{B}_{t,2}} \left[\nabla L^m_{\mathcal{B}}(\boldsymbol{w}^*) \right] \|^2 \right] \\
&= \mathbb{E}_{\mathcal{B}_{t,1}, \mathcal{B}_{t,2}, \boldsymbol{\epsilon}_r} \left[\|\nabla L^m_{\mathcal{B}}(\boldsymbol{w}^*) - \mathbb{E}_{\mathcal{B}_{t,1}, \mathcal{B}_{t,2}} \left[\nabla L^m_{\mathcal{B}}(\boldsymbol{w}^*) \right] \|^2 \right] + \mathbb{E}_{\boldsymbol{\epsilon}_r} \left[\|\mathbb{E}_{\mathcal{B}_{t,1}, \mathcal{B}_{t,2}} \left[\nabla L^m_{\mathcal{B}}(\boldsymbol{w}^*) \right] \|^2 \right] \\
&= \mathbb{E}_{\mathcal{B}_{t,1}, \mathcal{B}_{t,2}, \boldsymbol{\epsilon}_r} \left[\|\nabla L^m_{\mathcal{B}}(\boldsymbol{w}^*) - \mathbb{E}_{\mathcal{B}_{t,1}, \mathcal{B}_{t,2}} \left[\nabla L^m_{\mathcal{B}}(\boldsymbol{w}^*) \right] \|^2 \right] + \mathbb{E}_{\boldsymbol{\epsilon}_r} \left[\|\lambda \nabla L(\boldsymbol{w}^* + \boldsymbol{\epsilon}_r) - \lambda \nabla L^{\mathrm{Bayes}}(\boldsymbol{w}^*) \|^2 \right] \\
&\leq \frac{\sigma_0^2(2\lambda^2 - 2\lambda + 1)}{b} + \lambda^2 \sigma_m^*.
\end{aligned}$$

By developing squares, we obtain:

$$\|\boldsymbol{w}_{t+1} - \boldsymbol{w}^*\|^2 = \|\boldsymbol{w}_t - \boldsymbol{w}^*\|^2 - 2\gamma_t \langle \nabla L^m_{\mathcal{B}}(\boldsymbol{w}), \boldsymbol{w}_t - \boldsymbol{w}^* \rangle + \gamma_t^2 \|\nabla L^m_{\mathcal{B}}(\boldsymbol{w})\|^2.$$

Taking the expectation conditioned on $\boldsymbol{w}_t$ and using the convexity of $L^{\mathrm{m}}(\boldsymbol{w})$, we have:

$$\begin{aligned}
&\mathbb{E}_t \left[\|\boldsymbol{w}_{t+1} - \boldsymbol{w}^*\|^2 \right] \\
&= \|\boldsymbol{w}_t - \boldsymbol{w}^*\|^2 - 2\gamma_t \langle \nabla L^{\mathrm{m}}(\boldsymbol{w}_t), \boldsymbol{w}_t - \boldsymbol{w}^* \rangle + \gamma_t^2 \mathbb{E} \left[\|\nabla L^m_{\mathcal{B}}(\boldsymbol{w})\|^2 \right] \\
&\leq \|\boldsymbol{w}_t - \boldsymbol{w}^*\|^2 + 2\gamma_t \left(2\gamma_t \beta - 1 \right) \left(L^{\mathrm{m}}(\boldsymbol{w}_t) - L^{\mathrm{m}}(\boldsymbol{w}^*) \right) + 2\gamma_t^2 \left(\frac{2\lambda^2 - 2\lambda + 1}{b} \sigma_0^2 + \lambda^2 \sigma_m^* \right).
\end{aligned}$$

Taking expectation and summing over $k = 0, \cdots, t-1$ leads to:

$$\begin{aligned}
&2 \sum_{k=0}^{t-1} \gamma_k \left(1 - 2\gamma_k \beta \right) \mathbb{E} \left[L^{\mathrm{m}}(\boldsymbol{w}_k) - L^{\mathrm{m}}(\boldsymbol{w}^*) \right] \\
&\leq \|\boldsymbol{w}_0 - \boldsymbol{w}^*\|^2 - \mathbb{E} \left[\|\boldsymbol{w}_t - \boldsymbol{w}^*\|^2 \right] + 2 \left(\frac{2\lambda^2 - 2\lambda + 1}{b} \sigma_0^2 + \lambda^2 \sigma_m^* \right) \sum_{k=0}^{t-1} \gamma_k^2.
\end{aligned}$$

Dividing both sides by $2\sum_{k=0}^{t-1}\gamma_k\left(1-2\gamma_k\beta\right)$, we obtain:

$$\begin{aligned}&\sum_{k=0}^{t-1}\mathbb{E}\left[\frac{\gamma_k\left(1-2\gamma_k\beta\right)}{\sum_{k=0}^{t-1}\gamma_k\left(1-2\gamma_k\beta\right)}\left[L^{\mathrm{m}}(\boldsymbol{w}_k)-L^{\mathrm{m}}(\boldsymbol{w}^*)\right]\right]\\ \leq&\frac{\|\boldsymbol{w}_0-\boldsymbol{w}^*\|^2}{2\sum_{k=0}^{t-1}\gamma_k\left(1-2\gamma_k\beta\right)}+\frac{(\frac{2\lambda^2-2\lambda+1}{b}\sigma_0^2+\lambda^2\sigma_m^*)\sum_{k=0}^{t-1}\gamma_k^2}{\sum_{k=0}^{t-1}\gamma_k\left(1-2\gamma_k\beta\right)}.\end{aligned}$$

Hence, we arrive at the convergence result:

$$\begin{aligned}&\mathbb{E}\left[L(\bar{\boldsymbol{w}}^t)-L(\boldsymbol{w}^*)\right]\\ \leq&\frac{\|\boldsymbol{w}_0-\boldsymbol{w}^*\|^2}{2\sum_{k=0}^{t-1}\gamma_k\left(1-2\gamma_k\beta\right)}+\frac{(\frac{2\lambda^2-2\lambda+1}{b}\sigma_0^2+\lambda^2\sigma_m^*)\sum_{k=0}^{t-1}\gamma_k^2}{\sum_{k=0}^{t-1}\gamma_k\left(1-2\gamma_k\beta\right)}+\lambda\alpha\sigma\sqrt{d}\\ \leq&\frac{\|\boldsymbol{w}_0-\boldsymbol{w}^*\|^2}{2\gamma\sqrt{t}}+\frac{(\frac{2\lambda^2-2\lambda+1}{b}\sigma_0^2+\lambda^2\sigma_m^*)\gamma^2\log(t)}{\gamma\sqrt{t}}+\lambda\alpha\sigma\sqrt{d}\\ =&\frac{\|\boldsymbol{w}_0-\boldsymbol{w}^*\|^2}{2\gamma\sqrt{t}}+\frac{\gamma\log(t)}{\sqrt{t}}(\frac{2\lambda^2-2\lambda+1}{b}\sigma_0^2+\lambda^2\sigma_m^*)+\lambda\alpha\sigma\sqrt{d}.\end{aligned}$$

■

Appendix A2. Related Work

Flat Minima and Generalization. The connection between the flatness of local minima and generalization has been extensively studied [4, 17, 19, 22, 27]. Hochreiter *et al.* [14, 15] are among the first to reveal the connection between flat minima and the generalization of a model. Keskar *et al.* [22] observe that the performance degradation of large batch training is caused by converging to sharp minima. More recently, Jiang *et al.* [19] present a large-scale study of generalization in DNNs and demonstrate a strong connection between the sharpness and generalization error under various settings and hyper-parameters. Keskar *et al.* [22] and Dinh *et al.* [4] state that the flatness can be characterized by Hessian's eigenvalues and provide computationally feasible method to measure it.

Sharpness-aware Minimization (SAM). SAM [9] is a recently proposed training scheme that seeks flat minima by formulating a min-max problem and utilizing adversarial weight perturbation (AWP) to encourage parameters to sit in neighborhoods with uniformly low loss. It has shown power to achieve state-of-the-art performance. Later, a line of works improves the SAM's performance from the perspective of the neighborhood's geometric measure [23, 26, 29] or surrogate loss function [46]. Several methods have been developed to improve training efficiency [6, 7, 18, 28, 31, 43, 43].

Random Weight Perturbation (RWP). RWP is widely used in deep learning. Multiple weight noise injection methods have been shown to effectively escape spurious local optimum [45] and saddle points [20]. Upon generalization, Zhang *et al.* [44] discuss that RWP is much less effective for generalization improvement than AWP. Wen *et al.*. [37] propose SmoothOut framework to smooth out the sharp minima. Wang *et al.* [36] propose Gaussian model perturbation (GMP) as a regularization scheme for SGD training, but it remains inefficient due to the need of multiple computation budgets for noise sampling. Bisla *et al.* [2] connect the smoothness of the loss objective to generalization and

adopt filter-wise random Gaussian perturbation generation to improve the performance. However, the performance of RWP still lags behind that of AWP [2, 29]. Notably, recent Möllenhoff *et al.* [32] mathematically connect the expected Bayes loss under RWP with the min-max loss in SAM and suggest that RWP can be viewed as a 'softer' version of AWP. We significantly lift the performance of RWP from the convergence perspective and fill the performance gap to that of AWP.

Appendix A3. SAM and m-RWP Algorithm

Algorithm 1: SAM algorithm

Input: Loss function $L(\boldsymbol{w})$, training datasets $\mathcal{S} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^n$, initial weight $\boldsymbol{w}_{\text{init}}$, batch size b, neighborhood size ρ, learning rate γ

Output: Trained weight $\boldsymbol{w}$

Initialize weight $\boldsymbol{w} \leftarrow \boldsymbol{w}_{\text{init}}$;
while *not converged* **do**
  Sample a batch data $\mathcal{B}$ of size b from $\mathcal{S}$;
  Compute adversarial weight perturbation:
    $\boldsymbol{\epsilon}_s = \rho \frac{\nabla_{\mathcal{B}} L(\boldsymbol{w})}{\|\nabla_{\mathcal{B}} L(\boldsymbol{w})\|_2}$;
  Compute the gradient approximation $\boldsymbol{g}$:
    $\boldsymbol{g} \leftarrow \nabla L_{\mathcal{B}}(\boldsymbol{w} + \boldsymbol{\epsilon}_s)$;
  Update $\boldsymbol{w}$ using gradient descent:
    $\boldsymbol{w} \leftarrow \boldsymbol{w} - \gamma \boldsymbol{g}$;
end
return $\boldsymbol{w}$.

Algorithm 2: m-RWP algorithm

Input: Loss function $L(\boldsymbol{w})$, training datasets $\mathcal{S} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^n$, initial weight $\boldsymbol{w}_{\text{init}}$, batch size b, filter number k, noise variance σ, balance coefficient λ, learning rate γ

Output: Trained weight $\boldsymbol{w}$

Initialize weight $\boldsymbol{w} \leftarrow \boldsymbol{w}_{\text{init}}$;
while *not converged* **do**
  Sample batch data $\mathcal{B}_1$ and $\mathcal{B}_2$ of size b from $\mathcal{S}$;
  Generate random weight perturbations:
    $\boldsymbol{\epsilon}_r \sim \mathcal{N}\left(0, \sigma^2 diag\left(\|\boldsymbol{w}_1\|_2, \cdots, \|\boldsymbol{w}_k\|_2\right)\right)$;
  Compute the gradients $\boldsymbol{g}_1$ and $\boldsymbol{g}_2$ *in parallel*:
    $\boldsymbol{g}_1 \leftarrow \nabla L_{\mathcal{B}_1}(\boldsymbol{w})$, $\boldsymbol{g}_2 \leftarrow \nabla L_{\mathcal{B}_2}(\boldsymbol{w} + \boldsymbol{\epsilon}_r)$;
  Update $\boldsymbol{w}$ using gradient descent:
    $\boldsymbol{w} \leftarrow \boldsymbol{w} - \gamma(\lambda \boldsymbol{g}_1 + (1-\lambda)\boldsymbol{g}_2)$;
end
return $\boldsymbol{w}$.

Appendix A4. Magnitude Comparison of RWP and AWP

As the precise gradient direction is known, AWP is much more "effective" at perturbing the model compared to RWP. Consequently, in order to achieve a similar level of perturbation strength, the magnitude of RWP needs to be considerably larger than that of AWP. We carry out comparative experiments using a model $\boldsymbol{w}^*$ that has been well-trained with SGD and apply different magnitudes of perturbation for both RWP and AWP. As dipicted in Figure A1, to attain a similar expected perturbed training loss $\mathbb{E}\left[L(\boldsymbol{w}^* + \boldsymbol{\epsilon})\right]$, the perturbation radius of RWP would need to be two orders of magnitude larger than that of AWP. Such large perturbations can introduce instability in training and cause convergence issues that degrade the performance.

Appendix A5. Training Details

Training Settings. We compare the performance of four training schemes: SGD, SAM, RWP, and m-RWP. For CIFAR experiments, we set the training epochs to 200 with batch size 256, momentum 0.9, and weight decay 0.001 [6, 42], keeping the same among all schemes for a fair comparison (except for ViT-S, we adopt a longer 400 epochs training schedule with an initial learning rate of 0.0001 and Adam [24] as the base optimizer). We set ρ for SAM as 0.05 for CIFAR-10 and 0.10 for CIFAR-100 as in [9, 26]. For m-RWP/RWP, we try σ in $\{0.001, 0.005, 0.01, 0.015, 0.02, 0.03\}$ using ResNet-18 and finally use $\sigma = 0.015$ for m-RWP and $\sigma = 0.01$ for RWP for optimal. For ImageNet experiments, we set the training epochs to 90 with batch size 256, weight decay 0.0001,

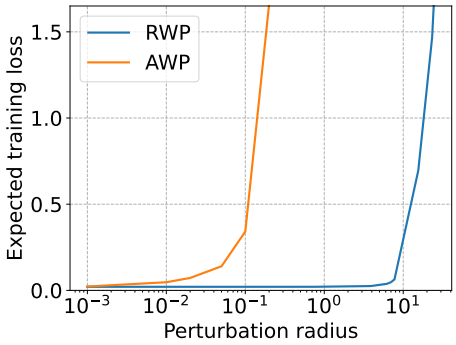
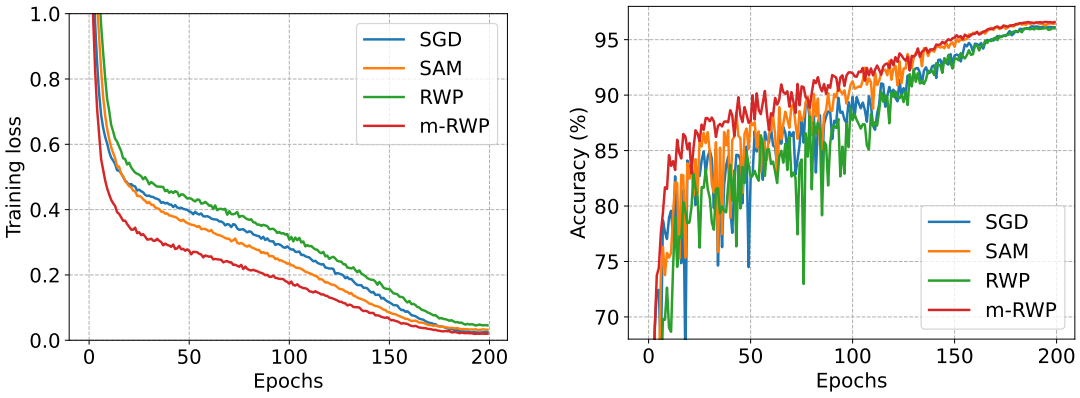


Figure A1: Expected training loss $\mathbb{E}[L(\boldsymbol{w}^* + \boldsymbol{\epsilon})]$ w.r.t. different perturbation radius $\|\boldsymbol{\epsilon}\|_2$ for RWP and AWP. The experiments employ a well-trained model $\boldsymbol{w}^*$ using SGD on CIFAR-10 with ResNet-18. Note that the x-axis is in logarithmic coordinates.

and momentum 0.9. We use $\rho = 0.05$ for SAM following [9, 26] and $\sigma = 0.005$, $\lambda = 0.5$ for m-RWP. We employ m-sharpness with $m = 128$ for SAM as in [9, 26]. For all experiments, we adopt cosine learning rate decay [30] with an initial learning rate of 0.1 and record the final model performance on the test set. Mean and standard deviation are calculated over three independent trials.

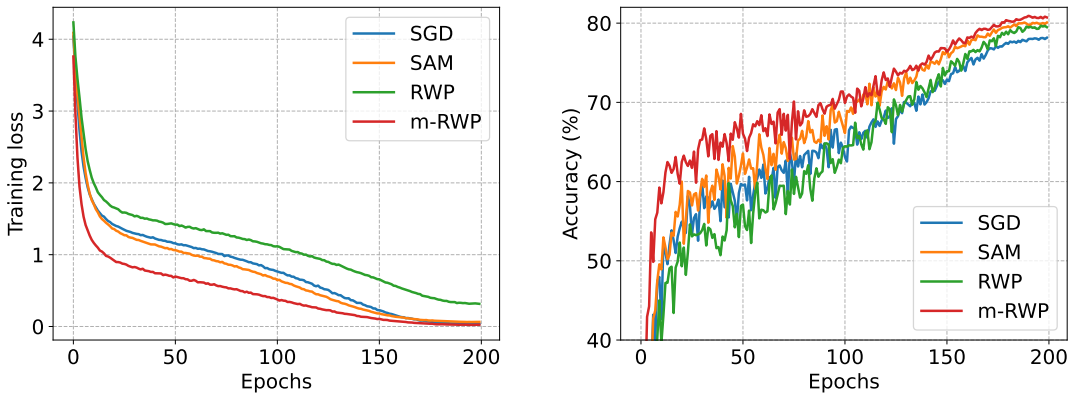
Appendix A6. Training Curves

Training Curves. We visualize the training curves of four training schemes (SGD, SAM, RMP, and m-RWP) using ResNet-18 on CIFAR-10 (Figure A2), CIFAR-100 (Figure A3), and ImageNet (Figure A4). Our results show that m-RWP achieves significantly faster convergence than the other competitive schemes in both training loss and test accuracy.



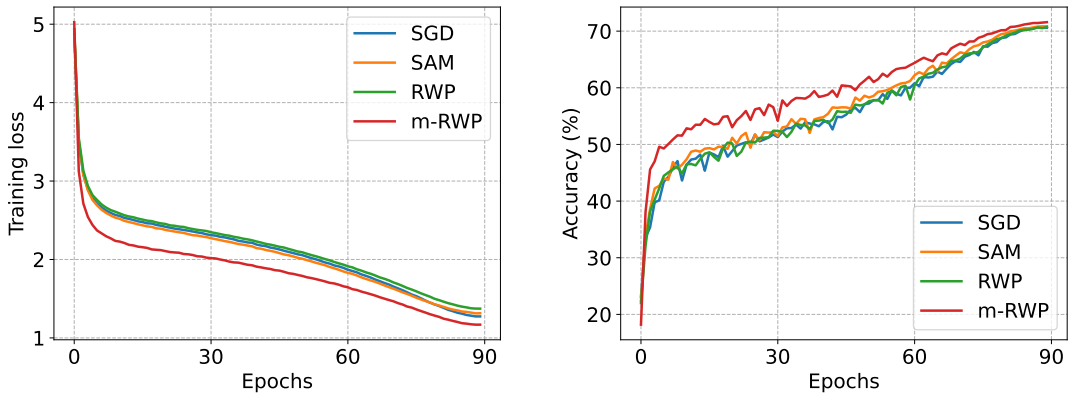
(*a*) Training loss (*b*) Test accuracy

Figure A2: Training curves on CIFAR-10 with ResNet-18 model.



(*a*) Training loss (*b*) Test accuracy

Figure A3: Training curves on CIFAR-100 with ResNet-18 model.



(*a*) Training loss (*b*) Test accuracy

Figure A4: Training curves on ImageNet with ResNet-18 model.

Appendix A7. Ablation Study and Visualization

Impact of different data batches. We further investigate the impact of same/different data batches for the two gradient steps in m-RWP and SAM. The results are presented in Table **??**. We observe that for m-RWP, the two choices yield comparable performance, with different batches being slightly better. However, for SAM, these two approaches are starkly different. Adopting different data batches would undesirably destroy the generalization performance of SAM to that of plain SGD. This finding suggests that applying the same data batch for adversarial attack and gradient propagation is a crucial part for SAM's generalization improvement. We attribute this to the unique characteristic of AWP in SAM, which is specific to a particular batch of data. *AWP over one batch can degenerate to meaningless noise w.r.t. another batch. In contrast, the*

Table A1: Effects of same/different data batches for two gradient steps. The experiments are conducted on CIFAR-100 with ResNet-18.

Training	Same	Different
SAM	80.22±0.23	78.30±0.11 (↓ 1.92)
m-RWP	81.04±0.10	81.25±0.13 (↑ 0.21)

perturbation for m-RWP is not associated to data and hence it allows using two different batches to accelerate convergence with better efficiency.

Sensitivity of hyper-parameters. m-RWP has two hyper-parameters, namely the noise magnitude σ and balance coefficient λ. To better understand their effects on performance, we test the performance under different choices of values. Specifically, we pick two representative network, i.e., VGG16-BN and ResNet-18, for CIFAR-10/100 datasets, and vary σ in $\{0.005, 0.01, 0.015, 0.02\}$ and λ in $\{0.1, 0.3, 0.5, 0.7, 0.9\}$. The results are in Figure A5. We observe that $\sigma = 0.015$ and $\lambda = 0.5$ is a robust choice for both architectures on CIFAR-10 and CIFAR-100.

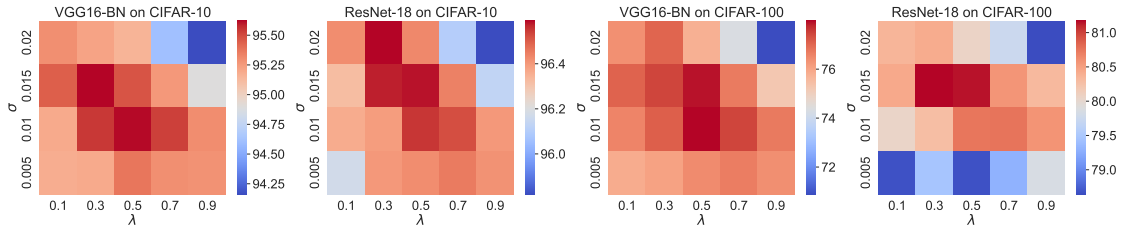


Figure A5: Performance under various hyper-parameter configurations.

Loss landscape and Hessian spectrum. Finally, we compare the loss landscape and Hessian spectrum of SAM and m-RWP. Following the plotting technique in [27], we uniformly sample 50×50 grid points in the range of $[-1, 1]$ from random "filter-normalized" direction [27], and for Hessian spectrum, we approximate it using the Lanczos algorithm [12]. In Figure A6, we observe that both methods could achieve flat loss landscape while m-RWP yields wider flat region with smaller dominant eigenvalue λ_1. This is perhaps due to the larger perturbations that m-RWP imposes. We thus conclude that m-RWP can converge to a flat minima as SAM does, or even better.

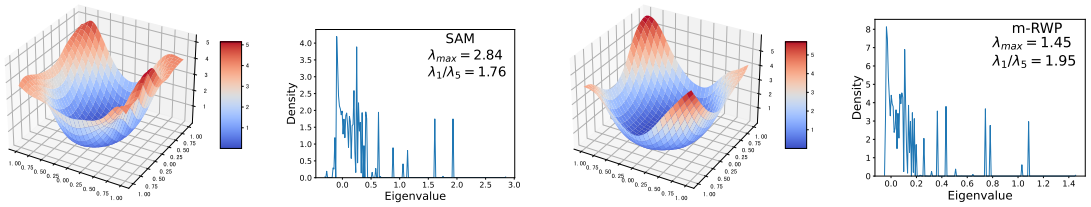


Figure A6: Loss landscape and Hessian spectrum visualization of SAM (**Left**) and m-RWP (**Right**). Models are trained on CIFAR-10 with ResNet-18.