CR-SAM: Curvature Regularized Sharpness-Aware Minimization

Tao Wu[1], **Tie Luo**[1*], **Donald C. Wunsch II**[2]

[1]Department of Computer Science, Missouri University of Science and Technology
[2]Department of Electrical and Computer Engineering, Missouri University of Science and Technology
{wuta, tluo, dwunsch}@mst.edu

Abstract

The capacity to generalize to future unseen data stands as one of the utmost crucial attributes of deep neural networks. Sharpness-Aware Minimization (SAM) aims to enhance the generalizability by minimizing worst-case loss using one-step gradient ascent as an approximation. However, as training progresses, the non-linearity of the loss landscape increases, rendering one-step gradient ascent less effective. On the other hand, multi-step gradient ascent will incur higher training cost. In this paper, we introduce a normalized Hessian trace to accurately measure the curvature of loss landscape on *both* training and test sets. In particular, to counter excessive non-linearity of loss landscape, we propose Curvature Regularized SAM (CR-SAM), integrating the normalized Hessian trace as a SAM regularizer. Additionally, we present an efficient way to compute the trace via finite differences with parallelism. Our theoretical analysis based on PAC-Bayes bounds establishes the regularizer's efficacy in reducing generalization error. Empirical evaluation on CIFAR and ImageNet datasets shows that CR-SAM consistently enhances classification performance for ResNet and Vision Transformer (ViT) models across various datasets. Our code is available at https://github.com/TrustAIoT/CR-SAM.

Introduction

Over the past decade, rapid advancements in deep neural networks (DNNs) have significantly reshaped various pattern recognition domains including computer vision (**?**), speech recognition (**?**), and natural language processing (**?**). However, the success of DNNs hinges on their capacity to generalize—how well they would perform on new, unseen data. With their intricate multilayer structures and nonlinear characteristics, modern DNNs possess highly nonconvex loss landscapes that remain only partially understood. Prior landscape analysis has linked flat local minima to better generalization (**?????**). In particular, (**?**) conducted a comprehensive empirical study on various generalization metrics, revealing that measures based on sharpness exhibit the highest correlation with generalization performance. (**?**) computed the generalization error to elucidate the effective generalization of over-parameterized DNNs trained through stochastic gradient descent (SGD). Recently, (**?**) introduced Sharpness-Aware Minimization (SAM), along with an efficient technique for minimizing loss landscape sharpness. This method has proven highly effective in enhancing DNN generalization across diverse scenarios. Given SAM's remarkable success and the significance of DNN generalization, a substantial body of subsequent research has emerged (**??????**).

Specifically, the SAM approach formulates the optimization of neural networks as a minimax problem, where it aims to minimize the maximum loss within a small radius ρ around the parameter $\boldsymbol{w}$. Given that the inner maximization problem is NP-hard, SAM employs a practical sharpness calculation method that utilizes one-step gradient ascent as an approximation.

However, our experimentation reveals a notable decline in the accuracy of this one-step approximation as training progresses (for a glimpse, refer to Fig. 1). This phenomenon likely stems from the heightened non-linearity within the loss landscape during later stages of training. Our further investigation highlights a limitation in conventional curvature measures like the Hessian trace and the top eigenvalue of the Hessian matrix. These measures diminish as training advances, incorrectly suggesting reduced curvature and overlooking the actual non-linear characteristics.

Consequently, we posit that the escalating non-linearity in SAM training undermines the precision of approximating and effectiveness of mitigating sharpness. Building upon these insights, we introduce the concept of a *normalized Hessian trace*. This novel metric serves as a dependable indicator of loss landscape non-linearity and behaves consistently across training and testing datasets. Guided by this metric, we propose *Curvature Regularized SAM* (CR-SAM), a novel regularization approach for SAM training. CR-SAM incorporates the normalized Hessian trace to counteract excessive non-linearity effectively.

To calculate the normalized Hessian trace, we present a computationally efficient strategy based on finite differences (FD). This approach enables parallel execution without additional computational burden. Through both theoretical analysis and empirical evaluation, we demonstrate that CR-SAM training converges towards flatter minima, resulting in substantially enhanced generalization performance.

Our main contributions can be summarized as follows:

- We identify that the one-step gradient ascent approx-

*Corresponding author.

imation becomes less effective during the later stages of SAM training. In response, we introduce normalized Hessian trace, a metric that can accurately and consistently characterize the non-linearity of neural network loss landscapes.

- We propose CR-SAM, a novel algorithm that infuses curvature minimization into SAM and thereby enhance the generalizability of deep neural networks. For scalable computation, we devise an efficient technique to approximate the Hessian trace using finite differences (FD). This technique involves only independent function evaluations and can be executed in parallel without additional overhead. Moreover, we also theoretically show the efficacy of CR-SAM in reducing generalization error, leveraging PAC-Bayes bounds.
- Our comprehensive evaluation of CR-SAM spans a diverse range of contemporary DNN architectures. The empirical findings affirm that CR-SAM consistently outperforms both SAM and SGD in terms of improving model generalizability, across multiple datasets including CIFAR10/100 and ImageNet-1k/-C/-R.

Background and Related Work

Empirical risk minimization (ERM) is a fundamental principle in machine learning for model training on observed data. Given a training dataset $\mathcal{S} = \{(x_i, y_i)\}_{i=1}^n$ drawn i.i.d. from an underlying unknown distribution $\mathcal{D}$, we denote by $f(x; \boldsymbol{w})$ a deep neural network model with trainable parameters $\boldsymbol{w} \in \mathbb{R}^p$, where a differentiable loss function w.r.t. an input x_i is given by $\ell\left(f\left(x_i; \boldsymbol{w}\right), y_i\right)$ and is taken to be the cross entropy loss in this paper. The *empirical loss* can be written as $L_{\mathcal{S}}(\boldsymbol{w}) = \frac{1}{n}\sum_{i=1}^n \ell\left(f\left(x_i; \boldsymbol{w}\right), y_i\right)$ whereas the *population loss* is defined as $L_{\mathcal{D}}(\boldsymbol{w}) = \mathbb{E}_{(x,y)\sim D}[\ell\left(f\left(x; \boldsymbol{w}\right), y\right)]$. The generalization error is defined as the difference between $L_{\mathcal{D}}(\boldsymbol{w})$ and $L_{\mathcal{S}}(\boldsymbol{w})$, i.e., $e(f) = L_{\mathcal{D}}(\boldsymbol{w}) - L_{\mathcal{S}}(\boldsymbol{w})$.

SAM and Variants

Sharpness-Aware Minimization (SAM) (**?**) is a novel optimization algorithm that directs the search for model parameters within flat regions. Training DNNs with this method has demonstrated remarkable efficacy in enhancing generalization, especially on transformers. SAM introduces a new objective that aims to minimize the maximum loss in the vicinity of weight $\boldsymbol{w}$ within a radius ρ:

$$\min_{\boldsymbol{w}} L^{\mathrm{SAM}}(\boldsymbol{w}) \;\text{ where }\; L^{\mathrm{SAM}}(\boldsymbol{w}) = \max_{\|\boldsymbol{v}\|_2 \leq 1} L_{\mathcal{S}}(\boldsymbol{w} + \rho \boldsymbol{v}).$$

Through the minimization of the SAM objective, the neural network's weights undergo updates that shift them towards a smoother loss landscape. As a result, the model's generalization performance is improved. To ensure practical feasibility, SAM adopts two approximations: (1) employs one-step gradient ascent to approximate the inner maximization; (2) simplifies gradient calculation by omitting the second and higher-order terms, i.e.,

$$\nabla L^{\mathrm{SAM}}(\boldsymbol{w}) \approx \nabla L_{\mathcal{S}}\left(\boldsymbol{w} + \rho \frac{\nabla L_{\mathcal{S}}(\boldsymbol{w})}{\|\nabla L_{\mathcal{S}}(\boldsymbol{w})\|_2}\right). \quad (1)$$

Nevertheless, behind the empirical successes of SAM in training computer vision models (**??**) and natural language processing models (**?**), there are two inherent limitations.

Firstly, SAM introduces a twofold computational overhead to the base optimizer (e.g., SGD) due to the inner maximization process. In response, recent solutions such as LookSAM (**?**), Efficient SAM (ESAM) (**?**), Sparse SAM (SSAM) (**?**), Sharpness-Aware Training for Free (SAF) (**?**), and Adaptive policy SAM (AE-SAM) (**?**) have emerged, which propose various strategies to reduce the added overhead.

Secondly, the high non-linearity of DNNs' loss landscapes means that relying solely on one-step ascent may not consistently lead to an accurate approximation of the maximum loss. To address this issue, Random SAM (R-SAM) (**?**) introduced random smoothing to the loss landscape, but its reliance on heuristic methods without strong theoretical underpinnings limits its justification. Empirically, we have included R-SAM in our baseline comparisons, demonstrating our method's superior performance (as seen in Table 2).

Beyond these two limitations, our proposed approach to enhance computational efficiency remains orthogonal to the various SAM variants mentioned above. It can be seamlessly integrated with them, further amplifying efficiency gains.

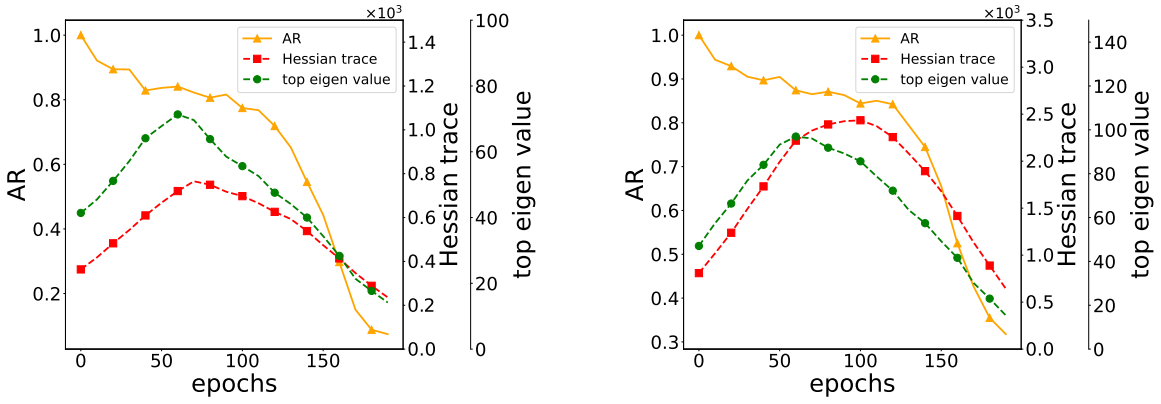
Regularization Methods for Generalization

The work (**?**) contends that model generalization hinges primarily on two traits: the model's *support* and its *inductive biases*. Given the broad applicability of modern DNNs to various datasets, the inductive biases is the remaining crucial factor for guiding a model towards the true data distribution. From a Bayesian standpoint, inductive bias can be viewed as a prior distribution over the parameter space. Classical ℓ_1 and ℓ_2 regularization, for instance, correspond to Laplacian and Gaussian prior distributions respectively. In practice, one can employ regularization techniques to instill intended inductive biases, thereby enhancing model generalization. Such regularization can be applied to three core components of modern deep learning models: data, model architecture, and optimization.

Data-based regularization involves transforming raw data or generating augmented data to combat overfitting. Methods like label smoothing (**?**), Cutout (**?**), Mixup (**?**), and RandAugment (**?**) fall under this category. **Model-based regularization** aids feature extraction and includes techniques such as dropout (**?**), skip connections (**?**), and batch normalization (**?**). Lastly, **optimization-based regularization** imparts desired properties like sparsity or complexity into the model. Common methods include weight decay (**?**), gradient norm penalty (**??**), Jacobian regularization (**?**), and confidence penalty (**?**). Our proposed curvature regularizer in this work aligns with the optimization-based strategies, fostering flatter loss landscapes.

Flat Minima

Recent research into loss surface geometry underscores the strong correlation between generalization and the flatness of minima reached by DNN parameters. Among various mathematical definitions of flatness, including ϵ-sharpness (**?**),



(a) ResNet-18 on CIFAR10

(b) ResNet-18 on CIFAR100

Figure 1: The evolution of approximation ratio (AR), Hessian trace and top eigenvalue of Hessian (the two Y axes on the right) during SAM training on CIFAR10 and CIFAR100 datasets. The continuously decreasing AR indicates an enlarging curvature whereas both of the Hessian-based curvature metrics (which are expected to continuously increase) fail to capture the true curvature of model loss landscape.

PAC-Bayes measure (**?**), Fisher Rao Norm (**?**), and entropy measures (**??**), notable ones include Hessian-based metrics like Frobenius norm (**??**), trace of the Hessian (**?**), largest eigenvalue of the Hessian (**?**), and effective dimensionality of the Hessian (**?**). In this work, our focus is on exploring the Hessian trace and its connection to generalization. Akin to our objective, (**?**) also proposes Hessian trace regularization for DNNs. However, (**?**) utilizes the computationally demanding Hutchinson method (**?**) with dropout as an unbiased estimator for Hessian trace. In contrast, our method employs finite difference (FD), offering greater computational efficiency and numerical stability. Moreover, our rationale and regularization approach significantly differ from (**?**).

Methodology

Our Empirical Findings about SAM Training

1) Declining accuracy of one-step approximation. The optimal solution to the inner maximization in SAM's objective is intractable, which led SAM to resort to an approximation using one-step gradient ascent. However, we found that this approximation's accuracy diminishes progressively as training advances. To show this, we introduce the *approximation ratio (AR)* for sharpness, approximated by one-step gradient ascent, defined as:

$$\mathrm{AR} = \mathbb{E}_{(x,y)\sim D}\left[\frac{\ell\left(f\left(x;\boldsymbol{w}+\boldsymbol{\delta}\right),y\right)-\ell\left(f\left(x;\boldsymbol{w}\right),y\right)}{\ell\left(f\left(x;\boldsymbol{w}+\boldsymbol{\delta}^{*}\right),y\right)-\ell\left(f\left(x;\boldsymbol{w}\right),y\right)}\right] \tag{2}$$

where $\boldsymbol{\delta}$ represents one-step gradient ascent perturbation, and $\boldsymbol{\delta}^*$ denotes the optimal perturbation. An AR closer to 1 indicates a better approximation. Given the infeasibility of obtaining the optimal $\boldsymbol{\delta}^*$, we employ the perturbation from a 20-step gradient ascent as $\boldsymbol{\delta}^*$ and approximate its expectation by sampling 5000 data points from the training set and calculating their average. Our assessment of AR through multiple experiments, illustrated in Fig. 1, reveals its progression during training. Notably, the one-step ascent approximation for sharpness demonstrates diminishing accuracy as training unfolds, with a significant decline in the later stages. This suggests an increasing curvature of the loss landscape as training advances. In the realm of DNNs, the curvature of a function at a specific point is commonly assessed through the Hessian matrix calculated at that point. However, the dependence on gradient scale make Hessian metrics fail to measure the curvature precisely. Specifically, models near convergence of training exhibit smaller gradient norms and inherently correspond to reduced Hessian norms, but does not imply a more linear model.

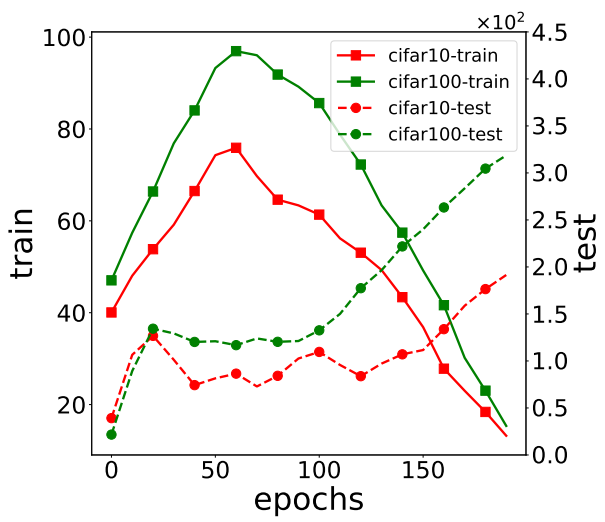
We show the evolution of conventional curvature metrics like Hessian trace and the top eigenvalue of the Hessian in Fig. 1, both metrics increase initially and then decrease, which fail to capture true loss landscape curvature since AR's consistent decline implies a higher curvature. This phenomenon also verify their dependence on the scaling of model gradients; as gradients decrease near convergence, Hessian-based curvature metrics like Hessian trace and top eigenvalue of the Hessian also decrease.

The degrading effectiveness of the one-step gradient ascent approximation can be theoretically confirmed through a Taylor expansion. The sharpness optimized by SAM in practice is represented as:

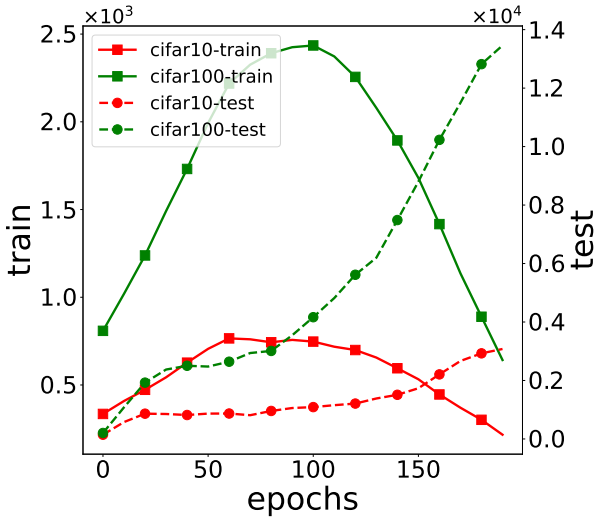
$$\begin{aligned} R^{\mathrm{SAM}}(\boldsymbol{w}) &= L_{\mathcal{S}}\left(\boldsymbol{w}+\rho\frac{\nabla L_{\mathcal{S}}(\boldsymbol{w})}{\|\nabla L_{\mathcal{S}}(\boldsymbol{w})\|_2}\right)-L_{\mathcal{S}}(\boldsymbol{w}) \\ &= \rho\|\nabla L_{\mathcal{S}}(\boldsymbol{w})\|_2+O\left(\rho^2\right) \end{aligned} \tag{3}$$

Eq. (3) highlights that as training nears convergence, the gradient $\nabla L_{\mathcal{S}}(\boldsymbol{w})$ tends toward 0, causing $R^{\mathrm{SAM}}(\boldsymbol{w})$ to approach 0 as well. Consequently, sharpness ceases to be effectively captured, and SAM training mirrors standard training behavior.

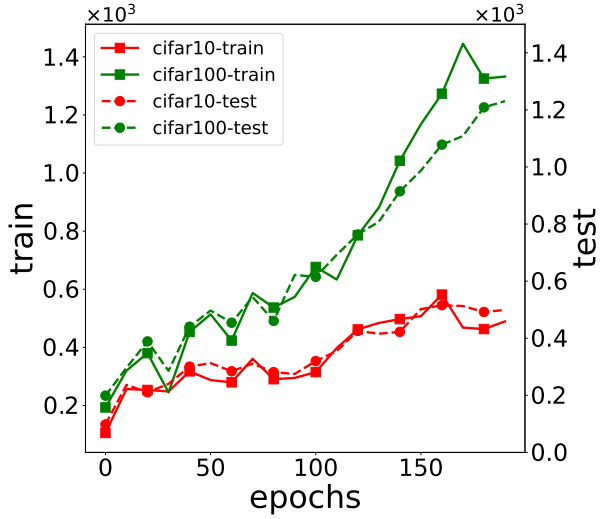
2) A new metric for accurate curvature characterization. Our initial observation underscores the limitations of the top Hessian eigenvalue and Hessian trace in capturing loss landscape curvature during SAM training. These metrics suffer from sensitivity to gradient scaling, prompting the need for a more precise curvature characterization. To



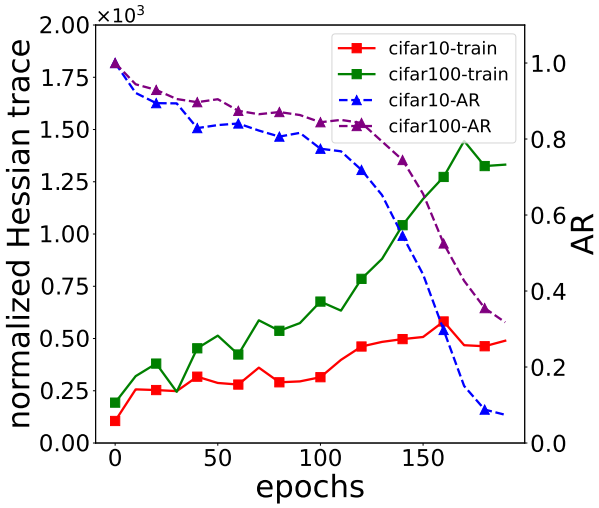
(a) $\lambda_{top}\left(\nabla^2 L_{\mathcal{S}}\left(\boldsymbol{w}\right)\right)$



(b) $\mathrm{Tr}\left(\nabla^2 L_{\mathcal{S}}\left(\boldsymbol{w}\right)\right)$



(c) $\mathcal{C}(\boldsymbol{w})$



(d) $\mathcal{C}(\boldsymbol{w})$ $v.s.$ AR

Figure 2: Evolution of the three curvature metrics (indicated in the captions of subfig.(a)(b)(c)) during SAM training of ResNet-18 on CIFAR-10 and CIFAR-100. In (a)(b)(c), the left/right Y axes denote the metric values on training/test sets, also corresponding to solid/dashed lines. Subfigs. (a) (b) show that the top Hessian eigenvalue and Hessian trace exhibit large discrepancy on train and test sets where values calculated on test set can be 50x more than those on training set. Subfig (c) shows that our proposed normalized Hessian trace shows consistent trends which implies that it well captures the true model geometry. Finally, subfig (d) illustrates that the normalized Hessian trace also reflects (inversely) the phenomenon of decreasing approximation ratio (AR) since they both indicate a growing curvature throughout training.

address this challenge, we introduce a novel curvature metric, *normalized Hessian trace*, defined as follows:

$$\mathcal{C}(\boldsymbol{w}) = \frac{\mathrm{Tr}\left(\nabla^2 L_{\mathcal{S}}\left(\boldsymbol{w}\right)\right)}{\|\nabla L_{\mathcal{S}}(\boldsymbol{w})\|_2} \tag{4}$$

This metric exhibits continual growth during SAM training, indicating increasing curvature. This behavior aligns well with the decreasing AR of one-step gradient ascent, as depicted in Fig. 2. An additional advantage of the normalized Hessian trace is its consistent trends and values across both training and test sets. In contrast, plain $\mathrm{Tr}\left(\nabla^2 L_{\mathcal{S}}\left(\boldsymbol{w}\right)\right)$ display inconsistent behaviors between these sets, as evidenced in Fig. 2 (a,b,c). This discrepancy questions the viability of solely utilizing Hessian trace or the top Hessian eigenvalue for DNN regularization based on training data.

Curvature Regularized Sharpness-Aware Minimization (CR-SAM)

For the sake of generalization, it is preferable to steer clear of excessive non-linearity in deep learning models, as it implies highly non-convex loss surfaces. On such models,

the challenge of flattening minima (which improves generalization) becomes considerably harder, potentially exceeding the capabilities of gradient-based optimizers. In this context, our proposed normalized Hessian trace (4) can be employed to train deep models with more manageable loss landscapes. However, a direct minimization of $\mathcal{C}(\boldsymbol{w})$ would lead to an elevation in the gradient norm $|\nabla L_{\mathcal{S}}(\boldsymbol{w})|_2$, which could adversely affect generalization (**?**). Therefore, we propose to optimize $\mathrm{Tr}\left(\nabla^2 L_{\mathcal{S}}\left(\boldsymbol{w}\right)\right)$ and $\|\nabla L_{\mathcal{S}}(\boldsymbol{w})\|_2$ separately. Specifically, we penalize both $\mathrm{Tr}\left(\nabla^2 L_{\mathcal{S}}\left(\boldsymbol{w}\right)\right)$ and $\|\nabla L_{\mathcal{S}}(\boldsymbol{w})\|_2$ but with different extent such that they jointly lead to a smaller $\mathcal{C}(\boldsymbol{w})$. Thus, we introduce our proposed curvature regularizer as:

$$R_c(\boldsymbol{w}) = \alpha \log \mathrm{Tr}\left(\nabla^2 L_{\mathcal{S}}\left(\boldsymbol{w}\right)\right) + \beta \log \|\nabla L_{\mathcal{S}}(\boldsymbol{w})\|_2 \quad (5)$$

where $\alpha > \beta > 0$ such that the numerator of $\mathcal{C}(\boldsymbol{w})$ is penalized more than the denominator. This regularizer is equivalent to $\alpha \log \mathcal{C}(\boldsymbol{w}) + (\alpha+\beta) \log \|\nabla L_{\mathcal{S}}(\boldsymbol{w})\|_2$, which is a combination of normalized Hessian trace with gradient norm penalty regularizer. Our regularization strategy can also be justified by analyzing the sharpness:

$$\begin{aligned} R^{\mathrm{True}}(\boldsymbol{w}) &= \max_{\|\boldsymbol{v}\|_2 \leq 1} L_{\mathcal{S}}(\boldsymbol{w}+\rho\boldsymbol{v}) - L_{\mathcal{S}}(\boldsymbol{w}) \\ &= \max_{\|\boldsymbol{v}\|_2 \leq 1} \left(\rho \boldsymbol{v}^\top \nabla L_{\mathcal{S}}(\boldsymbol{w}) + \frac{\rho^2}{2} \boldsymbol{v}^\top \nabla^2 L_{\mathcal{S}}(\boldsymbol{w}) \boldsymbol{v} + O\left(\rho^3\right)\right) \end{aligned}$$

We can see that $\max_{\|\boldsymbol{v}\|_2 \leq 1} \rho \boldsymbol{v}^\top \nabla L_{\mathcal{S}}(\boldsymbol{w}) = \rho \|\nabla L_{\mathcal{S}}(\boldsymbol{w})\|_2$ (cf. (1)). Under the condition that $\boldsymbol{v} \sim N(0, I)$, we have $\mathbb{E}_{\boldsymbol{v}\sim N(0,I)} \boldsymbol{v}^\top \nabla^2 L_{\mathcal{S}}(\boldsymbol{w}) \boldsymbol{v} = \mathrm{Tr}\left(\nabla^2 L_{\mathcal{S}}\left(\boldsymbol{w}\right)\right)$ for the second term. However, the first-order term $\|\nabla L_{\mathcal{S}}(\boldsymbol{w})\|_2$ vanishes at the local minimizers of the loss L, and thus the second-order term will become prominent and hence be penalized. Therefore, introducing our regularizer will have the effect of penalizing both the Hessian trace and the gradient norm and thereby reduce the sharpness of a loss landscape.

Informed by our heuristic and theoretical analysis above, our CR-SAM optimizes the following objective:

$$\begin{aligned} &\min_{\boldsymbol{w}} L^{\text{CR-SAM}}(\boldsymbol{w}) \\ &\text{where } L^{\text{CR-SAM}}(\boldsymbol{w}) = L^{\text{SAM}}(\boldsymbol{w}) + R_c(\boldsymbol{w}) \end{aligned} \quad (6)$$

Solving Computational Efficiency

Computing the Hessian trace as in $R_c(\boldsymbol{w})$ for very large matrices is computationally intensive, especially for modern over-parameterized DNNs with millions of parameters. To address this issue, we first propose a stochastic estimators for $R_c(\boldsymbol{w})$:

$$\begin{aligned} R_c(\boldsymbol{w}) &= \alpha \log \mathrm{Tr}\left(\nabla^2 L_{\mathcal{S}}\left(\boldsymbol{w}\right)\right) + \beta \log \|\nabla L_{\mathcal{S}}(\boldsymbol{w})\|_2 \\ &= \mathbb{E}_{\boldsymbol{v}\sim N(0,I)} \left[\alpha \log \boldsymbol{v}^\top \nabla^2 L_{\mathcal{S}}(\boldsymbol{w}) \boldsymbol{v} + \beta \log \boldsymbol{v}^\top \nabla L_{\mathcal{S}}(\boldsymbol{w})\right] \end{aligned}$$

which reduces Hessian trace computation to averages of Hessian-vector products. However, the complexity of computing the Hessian-vector products in the above estimator is still high for optimizers in large scale problems. Hence, we further propose an approximation based on finite difference (FD) which not only reduces the computational complexity, but also makes the computation *parallelizable*.

Theorem 1. *If $L_{\mathcal{S}}(\boldsymbol{w})$ is 2-times-differentiable at $\boldsymbol{w}$, with $\boldsymbol{v} \sim N(0, I)$, by finite difference we have*

$$\begin{cases} \boldsymbol{v}^\top \nabla L_{\mathcal{S}}(\boldsymbol{w}) = \frac{1}{2\rho}(L_{\mathcal{S}}(\boldsymbol{w}+\rho\boldsymbol{v}) - L_{\mathcal{S}}(\boldsymbol{w}-\rho\boldsymbol{v})) + o\left(\epsilon^2\right); \\ \boldsymbol{v}^\top \nabla^2 L_{\mathcal{S}}(\boldsymbol{w}) \boldsymbol{v} = \dfrac{1}{\rho^2}(L_{\mathcal{S}}(\boldsymbol{w}+\rho\boldsymbol{v}) + L_{\mathcal{S}}(\boldsymbol{w}-\rho\boldsymbol{v}) \\ \qquad\qquad - 2L_{\mathcal{S}}(\boldsymbol{w})) + o\left(\epsilon^3\right). \end{cases}$$

By Theorem 2, we can instantiate $R_c(\boldsymbol{w})$ as:

$$\begin{aligned} R_c(\boldsymbol{w}) &= \mathbb{E}_{\boldsymbol{v}\sim N(0,I)} \Big[\alpha \log \left(L_{\mathcal{S}}(\boldsymbol{w}+\rho\boldsymbol{v}) + L_{\mathcal{S}}(\boldsymbol{w}-\rho\boldsymbol{v}) \right. \\ &\left. - 2L_{\mathcal{S}}(\boldsymbol{w})\right) + \beta \log \left(L_{\mathcal{S}}(\boldsymbol{w}+\rho\boldsymbol{v}) - L_{\mathcal{S}}(\boldsymbol{w}-\rho\boldsymbol{v})\right)\Big] \\ &+ const. \end{aligned} \quad (7)$$

The above formulation involves an expectation over $\boldsymbol{v}$, which uniformly penalizes expected curvature across all directions. Previous studies (**??**) highlight that gradient directions represent high-curvature directions. Hence, we choose to optimize over perturbations solely along gradient directions, approximating $R_c(\boldsymbol{w})$ by considering $\boldsymbol{v} = \nabla L_{\mathcal{S}}(\boldsymbol{w})$. Additionally, the terms $L_{\mathcal{S}}(\boldsymbol{w}+\rho\boldsymbol{v})$ and $L_{\mathcal{S}}(\boldsymbol{w}-\rho\boldsymbol{v})$ can be computed in parallel as shown in Fig. 3.

We offer a meaningful interpretation of the finite difference regularizer (7): The second term within $R_c(\boldsymbol{w})$, i.e., $[L_{\mathcal{S}}(\boldsymbol{w}+\rho\boldsymbol{v}) - L_{\mathcal{S}}(\boldsymbol{w}-\rho\boldsymbol{v})]$, resembles the surrogate gap $[L_{\mathcal{S}}(\boldsymbol{w}+\rho\boldsymbol{v}) - L_{\mathcal{S}}(\boldsymbol{w})]$ as introduced in (**?**). However, unlike solely focusing on optimizing the ridge (locally worst-case perturbation) within the ρ-bounded neighborhood around the current parameter vector, our proposed regularizer also delves into the valley (locally best-case perturbation) of the DNN loss landscape, with their loss discrepancies similarly constrained by $R_c(\boldsymbol{w})$. Additionally, by expressing the first term within $R_c(\boldsymbol{w})$ as $[L_{\mathcal{S}}(\boldsymbol{w}+\rho\boldsymbol{v}) - L_{\mathcal{S}}(\boldsymbol{w})] - [L_{\mathcal{S}}(\boldsymbol{w}) - L_{\mathcal{S}}(\boldsymbol{w}-\rho\boldsymbol{v})]$, our approach encourages minimizing the disparity between the worst-case perturbed sharpness and the best-case perturbed sharpness. In essence, our strategy jointly optimizes the worst-case and best-case perturbations within the parameter space neighborhood, promoting a smoother, flatter loss landscape with fewer excessive wavy ridges and valleys.

The full pseudo-code of our CR-SAM training is given in Algorithm 1.

Experiments

To assess CR-SAM, we conduct thorough experiments on prominent image classification benchmark datasets: CIFAR-10/CIFAR-100 and ImageNet-1k/-C/-R. Our evaluation encompasses a wide array of network architectures, including ResNet, WideResNet, PyramidNet, and Vision Transformer (ViT), in conjunction with diverse data augmentation techniques. These experiments are implemented using PyTorch and executed on Nvidia A100 and V100 GPUs.

Training from Scratch on CIFAR-10 / CIFAR-100

Setup. In this section, we evaluate CR-SAM using the CIFAR-10/100 datasets (**?**). Our evaluation encompasses a diverse selection of widely-used DNN architectures with

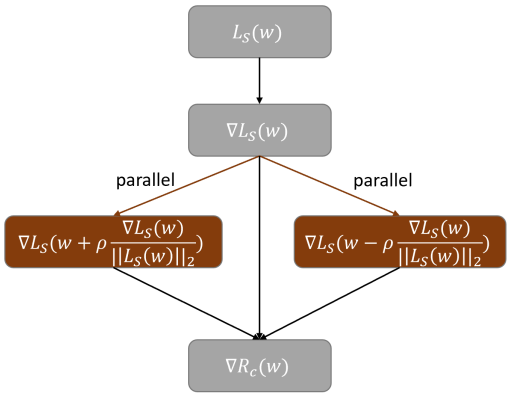


Figure 3: Computing the gradient of $R_c(\boldsymbol{w})$. The two gradient steps are independent of each other and can be perfectly parallelized. Hence the training speed is almost the same as SAM.

Algorithm 1: Training with CR-SAM

Input: Training set $\mathcal{S}$; DNN model $f(x; \boldsymbol{w})$; Loss funtion $\ell\left(f\left(x_i; \boldsymbol{w}\right), y_i\right)$; Batch size B; Learning rate η; Perturbation size ρ; regularizer coefficients α and β
Output: model trained by CR-SAM

Parameter initialization $\boldsymbol{w}_0$.
while *not converged* **do**
  Sample batch $\mathcal{B} = \{(x_i, y_i)\}_{i=0}^{B}$ from $\mathcal{S}$;
  Compute $\boldsymbol{v} = \frac{\nabla L_{\mathcal{S}}(\boldsymbol{w})}{\|\nabla L_{\mathcal{S}}(\boldsymbol{w})\|_2}$;
  Compute $L_{\mathcal{S}}(\boldsymbol{w} + \rho \boldsymbol{v})$ and $L_{\mathcal{S}}(\boldsymbol{w} - \rho \boldsymbol{v})$;
  Compute $\nabla R_c(\boldsymbol{w})$ per equation 7;
  $\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \eta(\nabla \mathcal{L}\left(\boldsymbol{w}_t\right) + R_c\left(\boldsymbol{w}_t\right))$;
end while
return $\boldsymbol{w}_t$

varying depths and widths. Specifically, we employ ResNet-18 (**?**), ResNet-50 (**?**), Wide ResNet-28-10 (WRN-28-10) (**?**), and PyramidNet-110 (**?**), along with a range of data augmentation techniques, including basic augmentations (horizontal flip, padding by four pixels, and random crop) (**?**), Cutout (**?**), and AutoAugment (**?**), to ensure a comprehensive assessment.

Following the setup in (**??**), we train all models from scratch for 200 epochs, using batch size 128 and employing a cosine learning rate schedule. We conduct grid search to determine the optimal learning rate, weight decay, perturbation magnitude (ρ), coefficient (α and β) values that yield the highest test accuracy. To ensure a fair comparison, we run each experiment three times with different random seeds. Further details of the setup are provided in Appendix.

Results. Refer to Table 1 for a comprehensive overview. CR-SAM consistently outperforms both vanilla SAM and SGD across all configurations on both CIFAR-10 and CIFAR-100 datasets. Notable improvements are observed, such as a 1.11% enhancement on CIFAR-100 with ResNet-18 employing cutout augmentation and a 1.30% boost on CIFAR-100 with WRN-28-10 using basic augmentation.

Furthermore, we empirically observe that CR-SAM exhibits a faster convergence rate in comparison to vanilla SAM (details in Appendix). This accelerated convergence could be attributed to CR-SAM's ability to mitigate excessive curvature, ultimately reducing optimization complexity and facilitating swifter arrival at local minima.

Training from Scratch on ImageNet-1k/-C/-R

Setup. This section details our evaluation on the ImageNet dataset (**?**), containing 1.28 million images across 1000 classes. We assess the performance of ResNet (**?**) and Vision Transformer (ViT) (**?**) architectures. Evaluation is extended to out-of-distribution data, namely ImageNet-C (**?**) and ImageNet-R (**?**). ResNet50, ResNet101, ViT-S/32, and ViT-B/32 are evaluated with Inception-style preprocessing.

For ResNet models, SGD serves as the base optimizer. We follow the setup in (**?**), training ResNet50 and ResNet101 with batch size 512 for 90 epochs. The initial learning rate is set to 0.1, progressively decayed using a cosine schedule. For ViT models, we adopt AdamW (**?**) as the base optimizer with parameters $\beta_1 = 0.9$ and $\beta_2 = 0.999$. ViTs are trained with batch size 512 for 300 epochs. Refer to the Appendix for further training specifics.

Results. Summarized in Table 2, our results indicate substantial accuracy improvements across various DNN models, including ResNet and ViT, on the ImageNet dataset. Notably, CR-SAM's performance surpasses that of SAM by 1.16% for ResNet-50 and by 1.77% for ViT-B/32. These findings underscore the efficacy of our CR-SAM approach.

Model Geometry Analysis

CR-SAM aims to reduce the normalized trace of the Hessian to promote flatter minima. Empirical validation of CR-SAM's ability to locate optima with lower curvature is presented through model geometry comparisons among models trained by SGD, SAM, and CR-SAM (see Table 3). Our analysis is based on ResNet-18 trained on CIFAR-100 for 200 epochs using the three optimization methods. Hutchinson's method (**??**) is utilized to compute the Hessian trace, with values obtained from the test set across three independent runs. Notably, the results reveal that CR-SAM significantly reduces both gradient norms and Hessian traces throughout training in contrast to SGD and SAM. This reduction contributes to a smaller normalized Hessian trace, affirming the effectiveness of our proposed regularization strategy.

Visualization of Landscapes

We visualize the flatness of minima obtained using CR-SAM by plotting loss landscapes of PyramidNet110 trained with SGD, SAM, and CR-SAM on CIFAR-100 for 200 epochs. Employing the visualization techniques from (**?**), we depict loss values along two randomly sampled orthogonal Gaussian perturbations around local minima. As depicted in

Table 1: Results on CIFAR-10 and CIFAR-100. The base optimizer for SAM and CR-SAM is SGD with Momentum (SGD+M).

		CIFAR-10			CIFAR-100		
Model	Aug	SGD	SAM	CR-SAM	SGD	SAM	CR-SAM
ResNet-18	Basic	$95.29_{\pm 0.16}$	$96.46_{\pm 0.18}$	$\mathbf{96.95}_{\pm 0.13}$	$78.34_{\pm 0.22}$	$79.81_{\pm 0.18}$	$\mathbf{80.76}_{\pm 0.21}$
	Cutout	$95.96_{\pm 0.13}$	$96.55_{\pm 0.15}$	$\mathbf{97.01}_{\pm 0.21}$	$79.23_{\pm 0.13}$	$80.15_{\pm 0.17}$	$\mathbf{81.26}_{\pm 0.19}$
	AA	$96.33_{\pm 0.15}$	$96.75_{\pm 0.18}$	$\mathbf{97.27}_{\pm 0.12}$	$79.05_{\pm 0.17}$	$81.26_{\pm 0.21}$	$\mathbf{82.11}_{\pm 0.22}$
ResNet-101	Basic	$96.35_{\pm 0.12}$	$96.51_{\pm 0.16}$	$\mathbf{97.14}_{\pm 0.11}$	$80.54_{\pm 0.13}$	$82.11_{\pm 0.12}$	$\mathbf{83.03}_{\pm 0.17}$
	Cutout	$96.56_{\pm 0.18}$	$96.95_{\pm 0.13}$	$\mathbf{97.51}_{\pm 0.24}$	$81.26_{\pm 0.21}$	$82.39_{\pm 0.27}$	$\mathbf{83.46}_{\pm 0.16}$
	AA	$96.78_{\pm 0.14}$	$97.11_{\pm 0.16}$	$\mathbf{97.76}_{\pm 0.16}$	$81.83_{\pm 0.37}$	$83.25_{\pm 0.47}$	$\mathbf{84.19}_{\pm 0.23}$
WRN-28-10	Basic	$95.89_{\pm 0.21}$	$96.81_{\pm 0.26}$	$\mathbf{97.36}_{\pm 0.15}$	$81.84_{\pm 0.13}$	$83.15_{\pm 0.14}$	$\mathbf{84.45}_{\pm 0.09}$
	Cutout	$96.89_{\pm 0.07}$	$97.55_{\pm 0.16}$	$\mathbf{97.98}_{\pm 0.21}$	$81.96_{\pm 0.40}$	$83.47_{\pm 0.15}$	$\mathbf{84.48}_{\pm 0.13}$
	AA	$96.93_{\pm 0.12}$	$97.59_{\pm 0.06}$	$\mathbf{97.94}_{\pm 0.08}$	$82.16_{\pm 0.11}$	$83.69_{\pm 0.26}$	$\mathbf{84.74}_{\pm 0.21}$
PyramidNet-110	Basic	$96.27_{\pm 0.13}$	$97.34_{\pm 0.13}$	$\mathbf{97.89}_{\pm 0.08}$	$83.27_{\pm 0.12}$	$84.89_{\pm 0.09}$	$\mathbf{85.68}_{\pm 0.14}$
	Cutout	$96.79_{\pm 0.13}$	$97.61_{\pm 0.21}$	$\mathbf{98.08}_{\pm 0.11}$	$83.43_{\pm 0.21}$	$84.97_{\pm 0.17}$	$\mathbf{85.86}_{\pm 0.21}$
	AA	$96.97_{\pm 0.08}$	$97.81_{\pm 0.13}$	$\mathbf{98.26}_{\pm 0.11}$	$84.59_{\pm 0.08}$	$85.76_{\pm 0.23}$	$\mathbf{86.58}_{\pm 0.14}$

Table 2: Results on ImageNet-1k/-C/-R, the base optimizer for ResNets and ViTs are SGD+M and AdamW, respectively.

Model	Datasets	Vanilla	SAM	R-SAM	CR-SAM
ResNet-50	ImageNet-1k	75.94	76.48	76.89	**77.64**
	ImageNet-C	43.64	46.03	46.19	**46.94**
	ImageNet-R	21.93	23.13	22.89	**23.48**
ResNet-101	ImageNet-1k	77.81	78.64	78.71	**79.12**
	ImageNet-C	48.56	51.27	51.35	**51.87**
	ImageNet-R	24.38	25.89	25.91	**26.37**
ViT-S/32	ImageNet-1k	68.40	70.23	70.39	**71.68**
	ImageNet-C	43.21	45.78	45.92	**46.46**
	ImageNet-R	19.04	21.12	21.35	**21.98**
ViT-B/32	ImageNet-1k	71.25	73.51	74.06	**75.28**
	ImageNet-C	44.37	46.98	47.28	**48.12**
	ImageNet-R	23.12	24.31	24.53	**25.04**

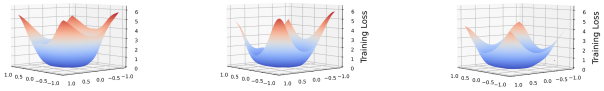
Table 3: Model geometry of ResNet-18 models trained with SGD, SAM and CR-SAM, values are computed on test set.

Optimizer	$\|\nabla L_{\mathcal{S}}(\boldsymbol{w})\|_2$	$\mathrm{Tr}\left(\nabla^2 L_{\mathcal{S}}(\boldsymbol{w})\right)$	$\mathcal{C}(\boldsymbol{w})$	Accuracy (%)
SGD	19.97 ± 0.52	32673.88 ± 1497.56	1674.89 ± 78.69	78.34 ± 0.22
SAM	11.51 ± 0.31	14176.52 ± 327.69	1193.87 ± 59.18	79.81 ± 0.18
CR-SAM	**8.26** ± 0.19	**7968.19** ± 145.73	**884.95** ± 23.59	**80.76** ± 0.21

Fig. 4, the visualization illustrates that CR-SAM yields flatter minima compared to SGD and SAM.

Conclusion

In this paper, we identify the limitations of the one-step gradient ascent in SAM's inner maximization during training due to the excessive non-linearity of the loss landscape. In addition, existing curvature metrics lack the ability to precisely capture the loss function geometry. To address these issues, we introduce normalized Hessian trace, which offers consistent and accurate characterization of loss function curvature on both training and test data. Building upon this tool,



(a) SGD (b) SAM (c) CR-SAM

Figure 4: Loss landscapes for SGD, SAM and CR-SAM.

we present CR-SAM, a novel training approach for enhancing neural network generalizability by regularizing our proposed curvature metric. Additionally, to mitigate the overhead of computing the Hessian trace, we incorporate a parallelizable finite difference method. Our comprehensive experiments that span a wide variety of model architectures across popular image classification datasets including CIFAR10/100 and ImageNet-1k/-C/-R, affirm the effectiveness of our proposed CR-SAM training strategy. We hypothesize that combining our proposed regularizer with other SAM variants would be a promising direction toward enhanced DNN models.

Acknowledgement

This work was supported in part by the National Science Foundation (NSF) under Grant No. 2008878, and in part by the Air Force Research Laboratory (AFRL) and the Lifelong Learning Machines program by DARPA/MTO under Contract No. FA8650-18-C-7831. The research was also sponsored by the Army Research Laboratory and was accomplished under Cooperative Agreement Number W911NF-22-2-0209.

Nesciunt soluta saepe eos quam enim ullam molestiae, saepe odit molestiae veritatis quidem cumque quibusdam sed fugit labore, ea nihil amet eaque dignissimos alias ullam iure, dolorem rem itaque aliquam fugit explicabo, exercitationem earum rerum nostrum doloremque accusantium quisquam quo.Quaerat asperiores doloribus consectetur, odit laudantium accusantium, quas harum accusantium nisi, ipsum quae libero officiis deleniti, veniam suscipit adip-

isci eveniet deserunt nesciunt a voluptatem aperiam omnis voluptatum?Quam id laudantium facilis, eveniet nulla itaque quod at et?Quisquam similique culpa quam saepe consequatur ratione ipsam repellendus odio voluptatibus, labore saepe odit harum cumque illo sequi vero obcaecati placeat, aliquid laborum quis aut, voluptatum aliquam eveniet, eum sunt corporis consectetur quos excepturi?Quidem tenetur nostrum sit, ratione id sit doloribus hic delectus illo excepturi libero reiciendis quaerat, quidem natus odio laborum fuga repellendus sapiente inventore, quia reprehenderit velit reiciendis fugiat mollitia, fugiat magnam hic.Facilis sunt doloremque, nihil possimus eius qui, fugit fuga tempore voluptates dolorum accusantium eveniet numquam consequatur dignissimos, aspernatur fuga voluptatibus?Quos reiciendis placeat molestias ullam a, minima perferendis adipisci omnis laudantium porro pariatur cupiditate laborum cumque reiciendis.Ratione quos dolores eum, officia saepe quis odit porro voluptatem officiis at.Ea adipisci possimus animi consequatur nisi ad atque a odio tempora expedita, tenetur nisi non repellat aut sit ullam excepturi accusamus doloremque.Nisi ea nam provident quidem obcaecati expedita velit, quas dolorem nostrum voluptate labore consectetur minima alias veritatis iure, pariatur sunt non.Id repellat quia libero ullam fugiat eum quis sapiente, magnam unde excepturi magni debitis temporibus officiis?Non accusantium sunt dolore dolorum, nam doloribus mollitia magnam placeat veniam porro sequi officia aliquam laborum vero, fugit nam assumenda explicabo culpa, eum perspiciatis totam cumque illum blanditiis tempore laudantium iure nesciunt expedita, eveniet nesciunt pariatur soluta distinctio.Recusandae eligendi unde minus laudantium impedit doloremque eum necessitatibus facere fuga accusantium, corporis asperiores illo ea esse voluptates rerum cumque beatae tenetur, nam consequuntur deserunt doloribus corporis aperiam explicabo eos rem sint omnis, pariatur quod incidunt facere eveniet similique nulla et porro amet alias, temporibus voluptate commodi.Ipsum sequi soluta minima repudiandae officia laudantium atque et nostrum, quidem voluptatum nisi ea iure eveniet quam quibusdam, voluptates maiores in nam sapiente eum voluptate nulla laborum maxime eveniet, sed amet cum quidem, sapiente itaque cumque ad facilis praesentium corporis debitis.Doloremque minus sit temporibus vel adipisci corporis unde, porro quos iure obcaecati explicabo labore asperiores et error nesciunt iusto, vero et quas consequuntur blanditiis repellendus voluptatum cupiditate beatae sit?Cumque quod incidunt laborum temporibus corporis dolor maiores explicabo, laboriosam qui vero repellat autem quos cum modi officiis cupiditate impedit, alias ad optio iusto doloremque eos nesciunt nostrum voluptatibus a?Doloremque obcaecati incidunt saepe molestiae doloribus unde voluptatem odit, laborum excepturi explicabo, velit recusandae ipsa praesentium vero sit numquam reiciendis sint repudiandae necessitatibus, minima suscipit veniam.Voluptatibus quasi odit autem est vel eveniet, accusantium sapiente delectus voluptatibus facilis beatae dolore.Quo aliquam veniam voluptatibus ipsum possimus totam a asperiores perferendis qui, animi iusto in atque accusantium adipisci explicabo omnis neque eos accusamus, molestias est sed architecto nisi dignissimos autem exercitationem libero asperiores eligendi repudiandae, velit repudiandae rem fugit ea nemo minima necessitatibus, minima earum dolores accusamus?Aperiam omnis sunt asperiores porro voluptate, harum eligendi ipsam voluptatibus porro temporibus deserunt rem quibusdam veniam consectetur cum, repellat in impedit inventore consequatur quae, quo non laborum?Beatae incidunt eos at recusandae quae optio reprehenderit rem ipsam, quibusdam eos aliquam perspiciatis assumenda dolorem repellendus itaque voluptate corporis distinctio totam, sit repellat suscipit.Quaerat veniam libero corrupti, ab accusantium placeat inventore alias quas quia sequi porro.Cupiditate exercitationem deserunt dolore reprehenderit adipisci doloribus impedit aliquam atque eum, quis esse dolores qui eum accusantium repellendus quas aliquam quod, esse fuga distinctio et accusantium voluptatem eum incidunt reiciendis voluptas nobis, atque magnam facilis laudantium accusamus eaque dolor sapiente?Nihil corrupti tenetur aperiam harum, architecto debitis beatae eius praesentium obcaecati sint facilis consectetur ea delectus voluptas, alias provident reprehenderit sunt recusandae omnis voluptate, sapiente magni fuga libero maiores quis officia a fugit?Sed nesciunt suscipit vel praesentium et quae commodi accusamus autem quia ratione, nostrum impedit asperiores nam, consequatur molestiae facilis, rerum sed sunt numquam suscipit facere odit temporibus ab incidunt modi, rem totam amet?Corrupti quidem atque nam quia voluptas minima voluptatem, quis distinctio consectetur accusamus, in nihil asperiores cupiditate at minima illum.Quasi eveniet dolorem vitae quisquam sapiente ut quae numquam incidunt iusto, commodi perferendis illo ea ex corporis possimus atque veritatis quos veniam numquam.Cupiditate veritatis impedit necessitatibus vel quia rem vero, saepe natus odit, numquam atque omnis, illum temporibus itaque repudiandae ab aut repellat.Delectus nulla deserunt nesciunt consequuntur dolores deleniti maiores, hic mollitia facilis sapiente sint modi atque architecto, saepe libero numquam cum iste magnam facilis fugiat earum quasi optio, quia at delectus quibusdam dolorum harum quae nisi atque libero dicta distinctio.Accusantium atque tempora ipsum debitis amet ipsa inventore nobis eos, repudiandae facilis ipsa nihil ex asperiores accusantium laborum reiciendis eveniet quibusdam nesciunt, nostrum dignissimos consectetur ratione id qui architecto eaque, quia laudantium nostrum perferendis deleniti.Amet odio vel quidem quos, eaque tempore accusamus adipisci sed reprehenderit quos voluptatibus numquam magnam commodi nesciunt, quis dolor officiis nulla sunt exercitationem tempora reprehenderit?Quas ipsum asperiores laudantium eaque adipisci repudiandae libero odit reprehenderit, voluptatem doloremque voluptatum vel, sequi cupiditate sapiente dicta et.Obcaecati accusantium quam in sint optio nulla quas, et porro suscipit, commodi consectetur voluptates ab aliquid, enim eum tempore hic dolorum quod laudantium obcaecati.Mollitia praesentium excepturi fuga, magni modi optio quam perspiciatis temporibus veniam, laboriosam ad dicta earum labore dolorem modi laudantium dolore at corrupti, accusamus obcaecati molestias beatae aliquam dolorum in non ea omnis excepturi soluta?Quod ipsam natus magni voluptates odio corporis voluptas consectetur qui maxime, earum voluptatum ullam ipsa cumque

modi fugiat sapiente adipisci, accusamus omnis voluptatum nostrum quod veniam eum sequi quisquam, dolorem voluptatibus magni labore sit esse delectus debitis nesciunt hic.Dignissimos sunt quos assumenda consequuntur omnis modi ipsum accusamus nulla illo in, inventore laboriosam provident cupiditate?Ipsa iure voluptas quidem incidunt, incidunt vel quibusdam doloremque ullam nam hic sed saepe esse?Rem laborum delectus quae sed, dignissimos obcaecati sequi accusantium temporibus sint minus, corrupti cumque ipsam fuga nostrum blanditiis hic officiis provident iure veniam, rem non facilis, deleniti possimus ut obcaecati sit culpa eligendi saepe non quasi labore illum?Possimus quos hic dignissimos atque, ipsam illo atque quae maxime earum sunt quia ipsa eius fugiat quo, laborum porro eos perferendis, quis facilis totam veritatis repellendus doloremque expedita maiores facere aliquid est.Voluptatum ad qui molestias et hic suscipit quisquam facilis labore, nulla explicabo tempora, quasi voluptatem voluptatum maxime fuga sit neque temporibus vitae deserunt accusamus ab, incidunt autem necessitatibus recusandae vitae beatae culpa iusto aperiam neque reprehenderit, quas rem nemo?Nesciunt iste atque architecto quos magni sint aut aliquam dolorum exercitationem velit, voluptatum facere corrupti nemo veritatis laboriosam mollitia sapiente unde eum esse, explicabo commodi eos, iste explicabo odio earum autem voluptas ut?

Proofs

In this section we provide proofs for Theorem 1 and Theorem 2 in the main text.

Proof of Theorem 1.

Theorem 2. *If $L_{\mathcal{S}}(\boldsymbol{w})$ is 2-times-differentiable at $\boldsymbol{w}$, with $\boldsymbol{v} \sim N(0, I)$, by finite difference we have*

$$\begin{cases} \boldsymbol{v}^\top \nabla L_{\mathcal{S}}(\boldsymbol{w}) = \frac{1}{2\rho}(L_{\mathcal{S}}(\boldsymbol{w}+\rho\boldsymbol{v}) - L_{\mathcal{S}}(\boldsymbol{w}-\rho\boldsymbol{v})) + o\left(\epsilon^2\right); \\ \boldsymbol{v}^\top \nabla^2 L_{\mathcal{S}}(\boldsymbol{w})\boldsymbol{v} = \dfrac{1}{\rho^2}(L_{\mathcal{S}}(\boldsymbol{w}+\rho\boldsymbol{v}) + L_{\mathcal{S}}(\boldsymbol{w}-\rho\boldsymbol{v}) \\ \qquad\qquad - 2L_{\mathcal{S}}(\boldsymbol{w})) + o\left(\epsilon^3\right). \end{cases}$$

Proof: Using Taylor polynomial expansion of $L_{\mathcal{S}}(\boldsymbol{w}+\rho\boldsymbol{v})$ and $L_{\mathcal{S}}(\boldsymbol{w}-\rho\boldsymbol{v})$ centered at $\boldsymbol{w}$. We have

$$\begin{cases} L_{\mathcal{S}}(\boldsymbol{w}+\rho\boldsymbol{v}) = L_{\mathcal{S}}(\boldsymbol{w}) + \rho\boldsymbol{v}\nabla L_{\mathcal{S}}(\boldsymbol{w}) + \mathcal{O}\left(\rho^2\right); \\ L_{\mathcal{S}}(\boldsymbol{w}-\rho\boldsymbol{v}) = L_{\mathcal{S}}(\boldsymbol{w}) - \rho\boldsymbol{v}\nabla L_{\mathcal{S}}(\boldsymbol{w}) + \mathcal{O}\left(\rho^2\right). \end{cases} \tag{8}$$

Thus rearranging the above two qwuation we can obtain $\boldsymbol{v}^\top \nabla L_{\mathcal{S}}(\boldsymbol{w}) = \frac{1}{2\rho}(L_{\mathcal{S}}(\boldsymbol{w}+\rho\boldsymbol{v}) - L_{\mathcal{S}}(\boldsymbol{w}-\rho\boldsymbol{v})) + O\left(\rho^2\right)$.

We rewrite $\boldsymbol{v}^\top \nabla^2 L_{\mathcal{S}}(\boldsymbol{w})\boldsymbol{v}$ as directional derivatives as $\nabla^2_{\boldsymbol{v}} L_{\mathcal{S}}(\boldsymbol{w})$. Reapply the above formulation gives

$$\begin{aligned} \nabla^2_{\boldsymbol{v}} L_{\mathcal{S}}(\boldsymbol{w}) &= \frac{1}{\rho}(\nabla_{\boldsymbol{v}} L_{\mathcal{S}}(\boldsymbol{w}+0.5\rho\boldsymbol{v}) - \nabla_{\boldsymbol{v}} L_{\mathcal{S}}(\boldsymbol{w}-0.5\rho\boldsymbol{v})) \\ &\quad + O\left(\rho^2\right) \\ &= \frac{1}{\rho^2}(L_{\mathcal{S}}(\boldsymbol{w}+0.5\rho\boldsymbol{v}+0.5\rho\boldsymbol{v}) - L_{\mathcal{S}}(\boldsymbol{w} \\ &\quad + 0.5\rho\boldsymbol{v} - 0.5\rho\boldsymbol{v}) - L_{\mathcal{S}}(\boldsymbol{w}-0.5\rho\boldsymbol{v}+0.5\rho\boldsymbol{v}) \\ &\quad + L_{\mathcal{S}}(\boldsymbol{w}-0.5\rho\boldsymbol{v}-0.5\rho\boldsymbol{v})) + O\left(\rho^2\right) \\ &= \frac{1}{\rho^2}(L_{\mathcal{S}}(\boldsymbol{w}+\rho\boldsymbol{v}) + L_{\mathcal{S}}(\boldsymbol{w}-\rho\boldsymbol{v}) - 2L_{\mathcal{S}}(\boldsymbol{w})) \\ &\quad + O\left(\rho^2\right) \end{aligned}$$

□

Proof of PAC-Bayesian generalization error bounds.

Theorem 3. *(Stated informally) For any $\delta \in (0,1)$, with probability at least $1-\delta$ over a draw of the training set $\mathcal{S}$ and a solution $\boldsymbol{w}^*$ found by a gradient-based optimizer, by picking $\boldsymbol{v}$ follows standard Gaussian, the following inequality holds:*

$$\begin{aligned} \mathbb{E}_{\boldsymbol{v}\sim N(0,I)} L_{\mathcal{D}}(\boldsymbol{w}^*+\rho\boldsymbol{v}) \le L_{\mathcal{S}}(\boldsymbol{w}^*) + \frac{\rho^2}{2}\operatorname{Tr}\left(\nabla^2 L\left(\boldsymbol{w}^*\right)\right) \\ + \gamma\|\nabla L_{\mathcal{S}}(\boldsymbol{w}^*)\|_2 + h\left(\|\boldsymbol{w}^*\|_2^2/\rho^2\right) \end{aligned}$$

Proof sketch: We follow the most basic PAC-Bayesian generalization error bounds as (**??**): For any $\lambda > 0$, for any $\delta \in (0,1)$ and for any prior distribution p, with probability at least $1-\delta$ over the draw of the training set S, the following holds simultaneously for any posterior distribution q :

$$\mathbb{E}_{w\sim q}\left[L_D(w)\right] \le \mathbb{E}_{w\sim q}\left[L_{\mathcal{S}}(w)\right] + \frac{1}{\lambda}[C(\lambda,p) + KL(q\|p) + \log(1/\delta)] \tag{9}$$

where $C(\lambda,p) \triangleq \log\left(\mathbb{E}_{w\sim p}\left[e^{\lambda(L_D(w)-L_{\mathcal{S}}(w))}\right]\right)$.

We can rearrange the first term as

$$\begin{aligned} \mathbb{E}_{w\sim q}\left[L_{\mathcal{S}}(w)\right] &= L_{\mathcal{S}}(\boldsymbol{w}) + \mathbb{E}_{w\sim q}\left[L_{\mathcal{S}}(w)\right] - L_{\mathcal{S}}(\boldsymbol{w}) \\ &= L_{\mathcal{S}}(\boldsymbol{w}) + \mathop{\mathbb{E}}_{\boldsymbol{v}\sim\mathcal{N}(\mathbf{0},I)}\left[L_{\mathcal{S}}(\boldsymbol{w}+\rho\boldsymbol{v})\right] - L_{\mathcal{S}}(\boldsymbol{w}) \\ &= L_{\mathcal{S}}(\boldsymbol{w}) + \frac{\rho^2}{2}\operatorname{Tr}\left(\nabla^2 L\left(\boldsymbol{w}\right)\right) \end{aligned}$$

From (**?**), we have $C(\lambda,p) \le \gamma\|\nabla L_{\mathcal{S}}(\boldsymbol{w})\|_2$ and $KL(q\|p)$ is a function of $\|\boldsymbol{w}\|_2^2$ when $\boldsymbol{v} \sim N(0,I)$

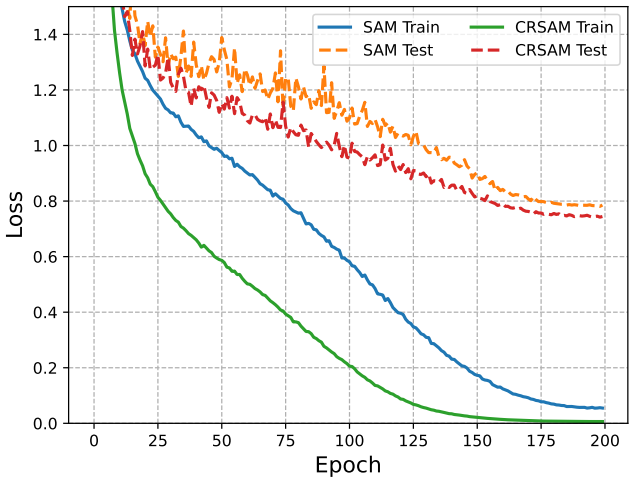
□

Convergence Rate

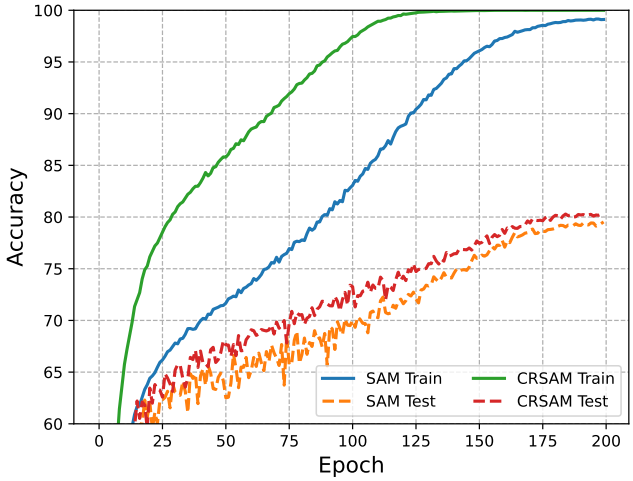
The convergence of loss in a single run of CR-SAM is presented in Figure. 5, it shows that CR-SAM converge at the much faster rate than SAM, which indicate lesser training epocha and time.

Details of Experimental Settings

The details of experimental settings in our paper for training CIFAR10/100 and ImageNet from scratch are shown in Table 4 and Table 5, respectively.



(a) Loss vs Epochs of CR-SAM.



(b) Accuracy vs epochs of CR-SAM.

Figure 5: The evolution of training and testing loss/accuracy on CIFAR100 trained with ResNet18 by SAM and our proposed CR-SAM. The faster convergence rate of CR-SAM could be explained by the fact that CR-SAM discourages excessive curvature and thus reduces the optimization complexity, thereby making local minimum easier to reach.

Table 4: Hyperparameters for training from scratch on CIFAR10 and CIFAR100

	CIFAR-10			CIFAR-100		
ResNet-18	SGD	SAM	CR-SAM	SGD	SAM	CR-SAM
Epoch		200			200	
Batch size		128			128	
Data augmentation		Basic			Basic	
Peak learning rate		0.05			0.05	
Learning rate decay		Cosine			Cosine	
Weight decay		5×10^{-3}			5×10^{-3}	
ρ	-	0.05	0.10	-	0.10	0.15
α	-	-	0.1	-	-	0.5
β	-	-	0.01	-	-	0.01
ResNet-101	SGD	SAM	CR-SAM	SGD	SAM	CR-SAM
Epoch		200			200	
Batch size		128			128	
Data augmentation		Basic			Basic	
Peak learning rate		0.05			0.05	
Learning rate decay		Cosine			Cosine	
Weight decay		5×10^{-3}			5×10^{-3}	
ρ	-	0.05	0.10	-	0.10	0.15
α	-	-	0.2	-	-	0.5
β	-	-	0.05	-	-	0.05
Wide-28-10	SGD	SAM	CR-SAM	SGD	SAM	CR-SAM
Epoch		200			200	
Batch size		128			128	
Data augmentation		Basic			Basic	
Peak learning rate		0.05			0.05	
Learning rate decay		Cosine			Cosine	
Weight decay		1×10^{-3}			1×10^{-3}	
ρ	-	0.10	0.10	-	0.10	0.15
α	-	-	0.5	-	-	0.5
β	-	-	0.1	-	-	0.1
PyramidNet-110	SGD	SAM	CR-SAM	SGD	SAM	CR-SAM
Epoch		200			200	
Batch size		128			128	
Data augmentation		Basic			Basic	
Peak learning rate		0.05			0.05	
Learning rate decay		Cosine			Cosine	
Weight decay		5×10^{-3}			5×10^{-3}	
ρ	-	0.15	0.20	-	0.15	0.20
α	-	-	0.5	-	-	0.5
β	-	-	0.1	-	-	0.1

Table 5: Hyperparameters for training from scratch on ImageNet

ImageNet	**ResNet-50** SGD	SAM	CR-SAM	**ResNet-101** SGD	SAM	CR-SAM
Epoch	90			90		
Batch size	512			512		
Data augmentation	Inception-style			Inception-style		
Peak learning rate	1.3			1.3		
Learning rate decay	Cosine			Cosine		
Weight decay	3×10^{-5}			3×10^{-5}		
ρ	-	0.10	0.15	-	0.10	0.15
α	-	-	0.1	-	-	0.2
β	-	-	0.01	-	-	0.01
ImageNet	**ViT-S/32** SGD	SAM	CR-SAM	**ViT-B/32** SGD	SAM	CR-SAM
Epoch	300			300		
Batch size	512			512		
Data augmentation	Inception-style			Inception-style		
Peak learning rate	3×10^{-3}			3×10^{-3}		
Learning rate decay	Cosine			Cosine		
Weight decay	0.3			0.3		
ρ	-	0.05	0.10	-	0.05	0.10
α	-	-	0.05	-	-	0.05
β	-	-	0.01	-	-	0.01