Asymptotic Unbiased Sample Sampling to Speed Up Sharpness-Aware Minimization

Jiaxin Deng[1], Junbiao Pang[1*], Baochang Zhang[2,3], Guodong Guo[4]

[1]School of Information Science and Technology, Beijing University of Technology, Beijing, China
[2]Hangzhou Research Institute, School of Artificial Intelligence, Beihang University, China
[3]Nanchang Institute of Technology, Nanchang, China
[4]Ningbo Institute of Digital Twin, Eastern Institute of Technology, Ningbo, China
dengjiaxin@emails.bjut.edu.cn, junbiao_pang@bjut.edu.cn, bczhang@buaa.edu.cn, guodong.guo@mail.wvu.edu

Abstract

Sharpness-Aware Minimization (SAM) has emerged as a promising approach for effectively reducing the generalization error. However, SAM incurs twice the computational cost compared to the base optimizer (*e.g.*, SGD). We propose Asymptotic Unbiased data sampling to accelerate SAM (AUSAM), which maintains the model's generalization capacity while significantly enhancing computational efficiency. Concretely, we probabilistically sample a subset of data points beneficial for SAM optimization based on a theoretically guaranteed criterion, *i.e.*, the Gradient Norm of each Sample (GNS). We further approximate the GNS by evaluating the difference in loss values before and after perturbation in SAM. As a plug-and-play, architecture-agnostic method, our approach consistently accelerates SAM across various tasks and networks, *i.e.,* classification, human pose estimation, and network quantization. On CIFAR-10/100 and Tiny-ImageNet, AUSAM achieves results comparable to SAM while providing a speedup of over 70%. By adjusting hyperparameters, AUSAM can match the speed of the base optimizer while significantly surpassing the base optimizer's performance. Compared to recent dynamic data pruning methods, AUSAM is better suited for SAM and excels in maintaining performance. Additionally, AUSAM accelerates optimization in human pose estimation and model quantization without sacrificing performance, demonstrating its broad practicality.

Introduction

The generalization capacity of a model significantly impacts its effectiveness and reliability in practical applications, making it a consistently important topic in deep learning research (Neyshabur et al. 2017; Izmailov et al. 2018; Cha et al. 2021; Andriushchenko and Flammarion 2022; Zhang et al. 2023). Several studies have delved into the correlation between generalization performance and the geometry of the loss landscape (Keskar et al. 2017; Jiang et al. 2020; Wu, Xia, and Wang 2020; Zhang et al. 2021). The loss landscape is complex and non-convex, with many local minima that exhibit varying generalization performance (Hochreiter and Schmidhuber 1994). These studies observed that flatter minima on the loss landscape tend to generalize more effectively than sharper minima. This motivates the idea of finding flatter minima during training.

Foret et al. recently introduced the Sharpness-Aware Minimization (SAM) (Foret et al. 2021) algorithm, which offers an elegant approach to seek flat minima and improve generalization. The SAM prevents the model from converging to sharp minima, thereby enhancing generalization. SAM and its variants have demonstrated state-of-the-art performance across various applications (Kwon et al. 2021; Jiang et al. 2023; Zhuang et al. 2022). However, SAM lacks efficiency due to its requirement of two forward and two backward in each optimization step. In recent years, various methods have been proposed to accelerate SAM. These can be roughly divided into methods for reducing gradient computations, such as G-RST (Zhao, Zhang, and Hu 2022), LookSAM (Liu et al. 2022), AE-SAM (Jiang et al. 2023), vSAM (Deng et al. 2024) and methods for efficient SAM optimization processes, such as SAF (Du et al. 2022b), K-SAM (Ni et al. 2022) and ESAM (Du et al. 2022a). However, most of them tends to potentially reduce the model's performance, especially the acceleration approach by reducing the number of SAM updates. Although some strategies can help models to maintain the performance, the extra operations introduced by implementing these strategies may limit gains in optimization efficiency.

In this paper, we introduce a dynamic sampling strategy for each mini-batch, based on maintaining expected sharpness across sampled and original data. Our theoretical analysis indicates that samples with larger gradient norms are crucial for preserving the model's generalization. Thus, we assign a score to each sample during forward propagation according to its gradient norm and prioritize higher-scoring samples in each mini-batch for training. This approach significantly improves optimization efficiency while preserving the model's generalization ability without excessive compromise. Compared to data pruning methods (Toneva et al. 2018; Paul, Ganguli, and Dziugaite 2021; Raju, Daruwalla, and Lipasti 2021), our method reduces the gradient expectation bias between the entire batch and the selected samples, achieving asymptotic unbiasedness as training progresses.

Our contributions are summarized as follows:

- We discovered that implementing SAM with mini-batch subsets characterized by high gradient norms effectively preserves the generalization capabilities of the model.

*Corresponding authors.

- We proposed an asymptotically unbiased data sampling strategy that accelerates the SAM optimization process without significantly compromising the model's generalization ability.

Related Work

Sharpness-Aware Minimization

Foret et al. introduced SAM (Foret et al. 2021), which aims to enhance the generalization ability of the model. SAM can be viewed as addressing a minimax optimization problem as follows:

$$\begin{aligned} &\min_{\mathbf{w}} L^{SAM}(\mathbf{w}) + \lambda||\mathbf{w}||^2 \\ &\quad \text{where } L^{SAM}(\mathbf{w}) = \max_{||\boldsymbol{\varepsilon}||\leq\rho} L(\mathbf{w}+\boldsymbol{\varepsilon}), \end{aligned} \tag{1}$$

where $\boldsymbol{\varepsilon}$ represents weight perturbations in an Euclidean ball with the radius ρ, $\lambda||\mathbf{w}||^2$ is the standard L2 regularization term. SAM utilizes Taylor expansion to search for the maximum perturbed loss in local parameter space:

$$\begin{aligned} \arg\max_{||\boldsymbol{\varepsilon}||\leq\rho} L(\mathbf{w}+\boldsymbol{\varepsilon}) &\approx \arg\max_{||\boldsymbol{\varepsilon}||\leq\rho} L(\mathbf{w}) + \boldsymbol{\varepsilon}^T\nabla_{\mathbf{w}}L(\mathbf{w}) \\ &= \arg\max_{||\boldsymbol{\varepsilon}||\leq\rho} \boldsymbol{\varepsilon}^T\nabla_{\mathbf{w}}L(\mathbf{w}). \end{aligned} \tag{2}$$

By solving Eq. (2), SAM can obtain the perturbation $\hat{\boldsymbol{\varepsilon}} = \rho\nabla_{\mathbf{w}}L(\mathbf{w})/||\nabla_{\mathbf{w}}L(\mathbf{w})||$ that maximizes the loss function $L^{SAM}(\mathbf{w})$. By substituting the perturbation back into Eq. (1) and the gradient approximation, the gradient during the weight optimization process in SAM is described as follows:

$$\nabla_{\mathbf{w}}L^{SAM}(\mathbf{w}) \approx \nabla_{\mathbf{w}}L(\mathbf{w}+\hat{\boldsymbol{\varepsilon}}) \approx \nabla_{\mathbf{w}}L(\mathbf{w})|_{\mathbf{w}+\hat{\boldsymbol{\varepsilon}}}. \tag{3}$$

From Eq. (2) and Eq. (3), we observe that SAM necessitates two forward and two backward to update weights once. This makes the computational overhead of SAM twice that of the basic optimizer.

Methods of Accelerating SAM

In this section, we discuss related work on methods for reducing gradient computations, methods for efficient SAM optimization processes, and dynamic data pruning methods.

Methods for reducing gradient computations. The motivation behind these methods is to reduce the number of times SAM optimization is utilized during the training. G-RST (Zhao, Zhang, and Hu 2022) performs a Bernoulli trial at each iteration, randomly selecting between SGD and SAM based on probabilities determined by a predefined scheduling function. LookSAM (Liu et al. 2022) employs SAM at every k iterations and reuses previous gradients in other iterations. AE-SAM (Jiang et al. 2023) adaptively employs SAM based on the loss landscape geometry. However, these methods often require extra operations to achieve optimization acceleration, which limits the overall speedup.

Methods for efficient SAM optimization processes. These methods aim to improve the efficiency of each step of SAM. SAF and MESA (Du et al. 2022b) employ the trajectory loss to approximate sharpness. However, SAF and MESA do not employ direct weight perturbations to help SAM escape local minimum regions. (Bahri, Mobahi, and Tay 2021) reveals SAM's benefits when data is limited by fine-tuning on sub-samples of the original task training split. But it randomly selects samples. K-SAM (Ni et al. 2022) proposes to compute gradients in both stages of SAM on only the top-k samples with highest loss. ESAM (Du et al. 2022a) uses fewer samples to compute the gradients and only updates part of the model in the SAM's second step. However, ESAM requires extra computation when selecting samples. In Momentum-SAM (Becker, Altrock, and Risse 2024), parameters are perturbed along the direction of the accumulated momentum vector, aiming to reduce sharpness and promote flatter minima. Although these methods improve the optimization efficiency, they may lead to a significant decrease in the generalization performance of the model.

Dynamic data pruning methods. Dynamic data pruning aims to reduce training costs by reducing the number of iterations for training (Toneva et al. 2018; Paul, Ganguli, and Dziugaite 2021). The pruning process is conducted during training, and sample information can be obtained during training. Dynamically prunes the samples based on easily attainable scores, e.g., loss values, during the training without trials. (Raju, Daruwalla, and Lipasti 2021) proposes three different data selection criteria in dynamic data pruning. Uncertainty sampling preferentially selects the sample with the lowest confidence in the current prediction of the model. ε-greedy approach is inspired by reinforcement learning. The upper-confidence bound (UCB) approach suggests pruning samples based on the upper confidence bound of their losses. InfoBatch (Qin et al. 2024) randomly prunes less informative samples based on loss distribution and rescales remaining samples' gradients to approximate the original gradient. However, these methods are not specific to SAM.

Method

Principle of Sampling

We aim to select a subset of samples from each mini-batch during training to reduce the computational costs of SAM's forward and backward propagation.

Theorem 1. *Suppose we select N samples from a mini-batch $\mathbf{B}$ of size K, denoting them as $\hat{\mathbf{B}}$ ($\hat{x_i} \in \hat{\mathbf{B}}$, $i = 1, 2, ..., N$), and denote the remaining set of $M(M = K - N)$ samples as $\check{\mathbf{B}}$ ($\check{x_i} \in \check{\mathbf{B}}$, $i = 1, 2, ..., M$). Let $\hat{\boldsymbol{\varepsilon}}_{\mathbf{B}}$ and $\hat{\boldsymbol{\varepsilon}}_{\hat{\mathbf{B}}}$ represent the parameter perturbations in SAM obtained through the set $\mathbf{B}$ and $\hat{\mathbf{B}}$, respectively. The difference in the Perturbation loss and Training loss between the calculations using a Full mini-batch (PTF) and those using Sampled samples (PTS) is as follows:*

$$\begin{aligned} &\left| \underbrace{[L_{\mathbf{B}}(\mathbf{w}+\hat{\boldsymbol{\varepsilon}}_{\mathbf{B}}) - L_{\mathbf{B}}(\mathbf{w})]}_{PTF} - \underbrace{[L_{\hat{\mathbf{B}}}(\mathbf{w}+\hat{\boldsymbol{\varepsilon}}_{\hat{\mathbf{B}}}) - L_{\hat{\mathbf{B}}}(\mathbf{w})]}_{PTS} \right| \\ &\leq \frac{\rho}{M}\sum_{i=1}^{M} \|\nabla_{\mathbf{w}}L_{\check{x_i}}(\mathbf{w})\| + \left|(O(\hat{\boldsymbol{\varepsilon}}_{\mathbf{B}}^2) - O(\hat{\boldsymbol{\varepsilon}}_{\hat{\mathbf{B}}}^2))\right|. \end{aligned} \tag{4}$$

When we ignore the error term, Theorem 1 demonstrates that the upper bound for the difference between PTF and PTS is smaller when the gradient norm of the unsampled samples is lower. One possible sampling method uses the gradient norm of each sample as the selection criterion. However, we still face a problem: how to efficiently compute the gradient norm of each sample $\|\nabla_{\mathbf{w}} L_{\mathbf{x}_i}(\mathbf{w})\|$ as the criterion?

Efficiently Per-sample gradient vs. Directly Approximate Gradient Norms

Efficiently computing per-sample gradient norm. The naive approach to computing per-sample gradient utilizes back-propagation when the mini-batch size is 1. However, this approach has significantly low optimization efficiency. (Goodfellow 2015) (Rochette, Manoel, and Tramel 2019) used the auto differentiation's intermediate results to compute the per-example gradient. However, these methods still require the extra operations to obtain the per-sample gradient. For instance, (Goodfellow 2015) has a time complexity of $O(Kmp^2)$ to compute the gradient of K samples, where m is the number of the layers in a neural network, and p is the number of dimensions in each layer. An extra time complexity of $O(K)$ is also incurred to compute gradient norms. A natural question is: is it possible to directly approximate per-sample gradient norms from the results of the perturbation in Eq. (2)?

Lemma 1. *For a mini-batch* $\mathbf{B}$*, the gradient at* $\mathbf{w}$ *is obtained as* $\nabla_{\mathbf{w}} L_{\mathbf{B}}(\mathbf{w})$ *and the parameter perturbation in SAM is* $\boldsymbol{\varepsilon}_{\mathbf{B}}$*. The training loss and perturbation loss of sample* $\mathbf{x}_i \in \mathbf{B}$ *denote as* $L_{\mathbf{x}_i}(\mathbf{w})$ *and* $L_{\mathbf{x}_i}(\mathbf{w}+\boldsymbol{\varepsilon}_{\mathbf{B}})$ *respectively. When the gradient* $\nabla_{\mathbf{w}} L_{\mathbf{B}}(\mathbf{w})$ *is nonzero, the gradient norm of sample* $\mathbf{x}_i$ *is approximated as follows:*

$$\begin{aligned}
&\|\nabla_{\mathbf{w}} L_{\mathbf{x}_i}(\mathbf{w})\| \\
&\approx \left\| \frac{L_{\mathbf{x}_i}(\mathbf{w} + \rho \frac{\nabla_{\mathbf{w}} L_{\mathbf{B}}(\mathbf{w})}{||\nabla_{\mathbf{w}} L_{\mathbf{B}}(\mathbf{w})||}) - L_{\mathbf{x}_i}(\mathbf{w})}{\rho} \frac{\nabla_{\mathbf{w}} L_{\mathbf{B}}(\mathbf{w})}{||\nabla_{\mathbf{w}} L_{\mathbf{B}}(\mathbf{w})||} \right\| \\
&= \frac{1}{\rho} \underbrace{|L_{\mathbf{x}_i}(\mathbf{w} + \boldsymbol{\varepsilon}_{\mathbf{B}}) - L_{\mathbf{x}_i}(\mathbf{w})|}_{DLP}.
\end{aligned} \tag{5}$$

Directly approximate gradient norm. Lemma 1 shows that the gradient norm of sample $\mathbf{x}_i$ can be approximated by the absolute values of **D**ifference in **L**oss before and after **P**erturbation (hereinafter referred to as DLP), *i.e.*, $|L_{\mathbf{x}_i}(\mathbf{w} + \boldsymbol{\varepsilon}_{\mathbf{B}}) - L_{\mathbf{x}_i}(\mathbf{w})|$. Therefore, instead of estimating the gradient norm $\|\nabla_{\mathbf{w}} L_{\mathbf{x}_i}(\mathbf{w})\|$, we use DLP as our sampling indicator. To improve the accuracy of the gradient norm evaluation, we estimate the current gradient norm by dynamically averaging the historical DLP for each sample. The Average DLP (ADLP) for sample $\mathbf{x}_i$ at epoch T is given by the following:

$$g_{\mathbf{x}_i}^T = \frac{1}{T-1} \sum_{t=1}^{T-1} |L_{\mathbf{x}_i}(\mathbf{w}_t + \boldsymbol{\varepsilon}_{\mathbf{B}_t}) - L_{\mathbf{x}_i}(\mathbf{w}_t)|. \tag{6}$$

The time and storage complexity of Eq. (6) are $O(1)$ and $O(D)$, respectively, where D is the number of samples in the dataset.

Sampling Strategy

To ensure that samples with low $g_{\mathbf{x}_i}^T$ values are not overlooked, we implement a probabilistic sampling mechanism. The sampling probability for each sample at epoch T is determined as follows:

$$p_{\mathbf{x}_i}^T = \frac{g_{\mathbf{x}_i}^T}{\sum_{i=1}^K g_{\mathbf{x}_i}^T}, \tag{7}$$

where $p_{\mathbf{x}_i}^T$ denotes the sampling probability of sample $\mathbf{x}_i$ at epoch T. To control the ratio of the maximum and minimum sampling probabilities, we normalize $g_{\mathbf{x}_i}^T$ using min-max normalization as follows:

$$\tilde{g}_{\mathbf{x}_i}^T = s_{min} + \frac{g_{\mathbf{x}_i}^T - g_{\mathbf{B}}^{min}}{g_{\mathbf{B}}^{max} - g_{\mathbf{B}}^{min}} (s_{max} - s_{min}), \tag{8}$$

where s_{min} and s_{max} represent the lower and upper bounds of the normalization, respectively. $g_{\mathbf{B}}^{max}$ and $g_{\mathbf{B}}^{min}$ are the maximum and minimum values of $g_{\mathbf{x}_i}^T$ within the mini-batch $\mathbf{B}$, respectively. Then, we calculate the probability $p_{\mathbf{x}_i}^T$ by substituting $g_{\mathbf{x}_i}^T$ with $\tilde{g}_{\mathbf{x}_i}^T$ in Eq. (7). In addition, we use a hyperparameter α to represent the proportion of samples selected from a mini-batch of size K, where αK samples are sampled. The time and storage complexity of this process are $O(K)$ and $O(1)$, respectively.

We gradually select samples with large gradient norms for stable initialization of training. We define a fixed epoch E_{start} and allow s_{max} to be a floating value before epoch E_{start}, linearly increasing from s_{min} to s_{max}. This enables samples with larger gradient norms to be increasingly prioritized for selection during training. Algorithm 1 shows the overall proposed algorithm.

Speed up ratio of SAM

Suppose that the time complexity of optimizing the model with SAM is $O(T)$. We can roughly divide the total time T into two parts: the forward and backward time T_f and the remaining time T_b. Our method can theoretically reduce the forward and backward time to αT_f. When T_b is sufficiently small, our method achieves nearly $O(0.5T)$ for $\alpha = 0.5$, which is almost as fast as the base optimizer.

Theoretical Analysis

Assumption 1. *(Smoothness).* $L(\mathbf{w})$ *is* τ*-Lipschitz smooth in* $\mathbf{w}$*, i.e.,* $\|\nabla L(\mathbf{w}) - \nabla L(\mathbf{v})\| \leq \tau \|\mathbf{w} - \mathbf{v}\|$.

Assumption 2. *(Bounded gradients). By the assumption that an upper bound is exists on the gradient of sampled set* $\hat{\mathbf{B}}$ *from each mini-batch. There exists* $G > 0$ *for* $\hat{\mathbf{B}}$ *such that* $\mathbb{E}\left[\left\|\nabla L_{\hat{\mathbf{B}}}(\mathbf{w})\right\|\right] \leq G$.

Assumption 3. *(Bounded variance of stochastic gradients). Given the training set* $\mathbf{D}$ *and a sampled set* $\hat{\mathbf{B}} \in \mathbf{D}$*. There exists* $\sigma \geq 0$*, the variance of the sampled set of size* αK *is bounded by* $\mathbb{E}\left[\left\|\nabla L_{\hat{\mathbf{B}}}(\mathbf{w}) - \nabla L_{\mathbf{D}}(\mathbf{w})\right\|^2\right] \leq \frac{\sigma^2}{\alpha K}$.

Algorithm 1: Pseudocode of the proposed method

Require: The training dataset $\mathbf{D}$, the learning rate η, the batch size K, parameters α, ρ, s_{min} and s_{max}.

1: **for** $t = 1, 2, \cdots$ **do**
2: Randomly sample a mini-batch $\mathbf{B}$ with size K;
3: Calculate the probability for each sample in set $\mathbf{B}$ of being selected according to Eq. (7).
4: Sampling αK data from $\mathbf{B}$ to constitute a subset $\hat{\mathbf{B}}$;
5: Calculate the loss for each selected sample $L_{\mathbf{x}_i}(\mathbf{w}_t)$ and gradient of all selected samples $\nabla L_{\hat{\mathbf{B}}}(\mathbf{w}_t)$.
6: Calculate perturbation $\hat{\varepsilon} = \rho \frac{\nabla L_{\hat{\mathbf{B}}}(\mathbf{w}_t)}{||\nabla L_{\hat{\mathbf{B}}}(\mathbf{w}_t)||}$ and gradient after perturbation $\nabla L_{\hat{\mathbf{B}}}(\mathbf{w}_t + \boldsymbol{\varepsilon})$.
7: Update the weights by $\mathbf{w}_{t+1} = \mathbf{w}_t - \eta \nabla L_{\hat{\mathbf{B}}}(\mathbf{w}_t + \boldsymbol{\varepsilon})$;
8: Update the ADLP for each selected sample based on Eq. (6).
9: **end for**

Theorem 2. *Suppose Assumption 1 and 2 hold. Let learning rate $\eta_t = \eta_0/t^2$, η_0 represents the initial learning rate, and there are N iterations in one epoch. The deviation between the average gradient norm of sample $\mathbf{x}_i$ over $T-1$ epochs and the gradient norm at the T-th epoch is bounded within a specific range, defined as follows:*

$$\left| \|\nabla L_{\mathbf{x}_i}(\mathbf{w}_T)\| - \frac{1}{T-1} \sum_{t=1}^{T-1} \|\nabla_{\mathbf{w}} L_{\mathbf{x}_i}(\mathbf{w}_t)\| \right| \leq \frac{\tau \eta_0 \pi^2 N G}{6}. \tag{9}$$

From Theorem 2, it is evident that the error is limited to a certain range. This suggests the viability of substituting the current gradient norm with the average gradient norm (Eq. 6).

Theorem 3. *Suppose Assumptions 1 and 3 hold. Let K be the mini-batch size, α is the sampling rate. For learn rate $\eta \leq \frac{1}{\tau}$ and $\rho \leq \frac{1}{4\tau}$, the algorithm satisfies:*

$$\begin{aligned} &\mathbb{E}\left[\sum_{t=1}^{T} \|\nabla L_{\mathbf{D}}(\mathbf{w}_t)\|^2\right] \\ &\leq \frac{16}{7\eta T}\left(L[(\mathbf{w}_0)] - E[L(\mathbf{w}_T)]\right) + \frac{17\eta\tau\sigma^2}{7\alpha K}. \end{aligned} \tag{10}$$

From Theorem 3, it has been observed that convergence in sample selection for training remains feasible, with tighter bounds resulting from selecting more samples from each mini-batch.

Theorem 4. *Assuming sample $\mathbf{x}$ from $\mathbf{D}$ are drawn from continuous distribution $q(\mathbf{x})$. $p_{\mathbf{x}}^t$ is the probability that sample $\mathbf{x}$ is sampled in the t-th epoch. The perturbation of SAM is assumed to be the theoretical perturbation over the entire dataset $\mathbf{D}$, i.e. $\boldsymbol{\varepsilon}_{\mathbf{D}} = \rho \nabla_{\mathbf{w}} L_D(\mathbf{w}_t)//\|\nabla_{\mathbf{w}} L_D(\mathbf{w}_t)\|$. The expected difference between the perturbation loss and the training loss on dataset $\mathbf{D}$ and its sampled subsets $\hat{\mathbf{D}}$ can be expressed as:*

$$\begin{aligned} &\left| \mathop{\mathbb{E}}_{\mathbf{x}\in\mathbf{D}} \left[\max_{\|\boldsymbol{\varepsilon}_{\mathbf{D}}\|\leq\rho} [L_{\mathbf{x}}(\mathbf{w}_t + \boldsymbol{\varepsilon}_{\mathbf{D}}) - L_{\mathbf{x}}(\mathbf{w}_t)] \right] \right. \\ &\qquad \left. - \mathop{\mathbb{E}}_{\mathbf{x}\in\hat{\mathbf{D}}} \left[\max_{\|\boldsymbol{\varepsilon}_{\mathbf{D}}\|\leq\rho} [L_{\mathbf{x}}(\mathbf{w}_t + \boldsymbol{\varepsilon}_{\mathbf{D}}) - L_{\mathbf{x}}(\mathbf{w}_t)] \right] \right| \\ &\leq \rho \int_{\mathbf{x}} (1 - p_{\mathbf{x}}^t) \|\nabla_{\mathbf{w}} L_{\mathbf{x}}(\mathbf{w}_t)\| \, q(\mathbf{x}) d\mathbf{x}. \end{aligned} \tag{11}$$

During the stage where SAM finds the internal maximum, Theorem 4 indicates that if we choose samples with larger gradient norms $\|\nabla_{\mathbf{w}} L_{\mathbf{x}}(\mathbf{w}_t)\|$ with a higher probability $p_{\mathbf{x}}^t$, the expected error generated by these samples will be smaller compared to randomly selected samples. As t increases, the gradient norm of sample $\mathbf{x}$ will theoretically gradually decrease, and the expected error will also gradually decrease. We still need to use these sampled samples during the SAM gradient descent phase, as they help identify the maximum loss within the parametric neighborhood space. Also, as t increases, samples with smaller gradient norms become less influential. Therefore, we choose samples with larger gradient norms to better represent the full mini-batch.

Experimental Results

Setup

Datasets and Models. We conduct experiments on the CIFAR-10, CIFAR-100 (Krizhevsky, Hinton et al. 2009) and Tiny-ImageNet (Le and Yang 2015) image classification benchmark datasets. We employ a variety of architectures, i.e., ResNet-18 (He et al. 2016), WideResNet-28-10 (Zagoruyko and Komodakis 2016) and PyramidNet-110 (Han, Kim, and Kim 2017) on CIFAR-10 and CIFAR-100, ResNet-18, ResNet-50 (He et al. 2016), and MobileNetv1 (Howard et al. 2017) on Tiny-ImageNet (Chrabaszcz, Loshchilov, and Hutter 2017), to evaluate the performance and training efficiency.

Baselines. We use vanilla SGD and SAM (Foret et al. 2021) as baselines. To comprehensively evaluate performance, we have also chosen some efficient methods ESAM (Du et al. 2022a), LookSAM (Liu et al. 2022), SAF (Du et al. 2022b), MESA (Du et al. 2022b), K-SAM (Ni et al. 2022) for comparison. These methods are the follow-up works of SAM that aim to enhance efficiency. Additionally, we choose InforBatch (Qin et al. 2024), an advanced dynamic data pruning method, and apply it to SAM. We also establish a dynamic pruning baseline, referred to as "SAM+Random", which randomly selects half of the samples from each batch.

Implementation Details. We trained all the models for 200 epochs with a batch size of 128, employing cutout regularization (DeVries and Taylor 2017) and cosine learning rate decay (Loshchilov and Hutter 2016) for all methods. For the proposed method, we set $s_{min} = 0.1$, $s_{max} = 0.5$ for ResNet-18 and WideResNet-28-10, $s_{max} = 1$ for PyramidNet-110. Typically, s_{max} is set to 1, though for smaller networks, it may be reduced further. This adjustment is motivated by the heightened impact of omitting samples with small gradient norms on smaller networks. Lowering

Methods	Resnet-18		Wideresnet-28-10	
	Accuracy	SAM%	Accuracy	SAM%
SGD	79.84±0.45	171%	82.47±0.30	195%
SAM	**81.04±0.29**	100%	84.71±0.21	100%
AUSAM-0.4	80.14±0.29	155%	84.44±0.31	202%
AUSAM-0.5	80.67±0.05	132%	84.62±0.26	179%
AUSAM-0.6	80.69±0.15	119%	**84.73±0.29**	152%
AUSAM-0.7	80.92±0.17	114%	84.66±0.07	134%

Table 1: Parameter Study of α. The best accuracy is in bold and the second best is underlined.

s_{max} increases the chances for samples with smaller gradient norms to be chosen.

For Tiny-ImageNet, we also train ResNet-18, ResNet-50 and MobileNetv1 (Howard et al. 2017) for 200 epochs using a batch size of 128 with cutout and cosine learning rate decay. For ResNet-18, ResNet-50, we use 64×64 resolution images and transform the first convolution of the models to a 3×3 convolution. For MobileNetv1, we resize the original images to 224×224.

We rename AUSAM as AUSAM-α according to the value of α when reporting the results. Each experiment was repeated three times using different random seeds to calculate the standard deviation, and the results are reported as average. SAM% represents the ratio of the training speed of a method to that of SAM. We implement AUSAM in Pytorch and train models on a single NVIDIA GeForce RTX 3090.

Parameter Studies

We study the effect of α on accuracy and optimization efficiency using ResNet-18 and WideResNet-28-10 on the CIFAR-100 dataset. The corresponding results are summarized in Tab. 1.

α determines the number of samples in each mini-batch, significantly influencing both the efficiency and accuracy of model training. As α decreases, the acceleration ratio of AUSAM increases gradually. This is because, as the number of samples selected in each mini-batch decreases, forward and backward propagation becomes more and more efficient. In theory, when $\alpha = 0.5$, the speed of AUSAM can nearly match that of the base optimizer, such as SGD. However, the time spent on other operations, such as data loading, is non-negligible compared to the time spent on forward and backward, AUSAM cannot accelerate these operations. Therefore, it cannot achieve the same speed as SGD. For ResNet-18, due to the smaller model size, the optimization time constitutes a smaller proportion of the overall training time, leading to a relatively low overall acceleration ratio. For WideResNet, the network requires more time for forward and backward, which allows AUSAM to fully utilize its performance. When $\alpha = 0.4$, the training speed matches that of SGD, and the performance is significantly improved compared to SGD.

Comparison to SOTAs

The experimental results on CIFAR-10 and CIFAR-100 are presented in Tab. 2. We observe that AUSAM achieves significantly higher accuracy compared to SGD. Furthermore, in comparison to LookSAM and ESAM, our results outperform theirs in most cases. This demonstrates that AUSAM can successfully preserve the model's generalization ability during the training process. This is probably because AUSAM essentially reduces the batch size, which helps to improve the generalization of the model (He, Liu, and Tao 2019). Although the total number of iterations for the entire training process did not increase, the selected samples accelerated the model's training. AUSAM also achieves superior results compared to SAF and MESA. This is because SAF and MESA optimize sharpness based on multiple previous iterations up to the current one, which differs from the sharpness optimization at the current iteration alone.

When $\alpha = 0.5$, these experiments demonstrate that AUSAM exhibits about 70% faster training than SAM while achieving comparable accuracy. For ResNet-18, the benefits of AUSAM are not fully apparent due to the inherently shorter duration required for both forward and backward during the optimization process. With larger models like WideResNet-28-10 and PyramidNet-110, AUSAM's benefits become more pronounced, resulting in faster training. When $\alpha = 0.4$, for WideResNet-28-10 and PyramidNet-110, AUSAM's speed is close to that of the base optimizer (SGD), and its performance is superior to the base optimizer (SGD). Compared to SAF and K-SAM, AUSAM performs even better.

The experimental results on Tiny-ImageNet are presented in Tab. 3. Due to the larger image resolution of Tiny-ImageNet compared to CIFAR-10 and CIFAR-100, the network requires more time for both forward and backward computations, leading to a higher acceleration ratio for AUSAM. With $\alpha = 0.5$, AUSAM achieves near SGD speed while sustaining superior accuracy in comparison to SGD for these three models. On ResNet-18 and MobileNetv1, AUSAM achieves results comparable to SAM.

Comparison to Dynamic Data Pruning

Data pruning aims to achieve lossless performance while minimizing overall costs by pruning less informative data at each epoch and training with the remaining data. We apply the advanced dynamic data pruning method InfoBatch (Qin et al. 2024) to the CIFAR-100 and conduct experiments using SGD and SAM. SAM+Info* represents the use of InfoBatch during the first loss calculation in SAM, while SAM+Info indicates its use during the second loss calculation in SAM. The results are presented in Tab. 5. Although InfoBatch improves optimization efficiency, it does not guarantee lossless optimization performance for SAM on CIFAR-100. Most dynamic data pruning methods struggle to effectively leverage the two losses before and after perturbation in SAM. Typically, these methods select samples with larger training losses directly, but such samples may not necessarily have larger losses in the parameter neighborhood space, which contradicts SAM's optimization principle.

	CIFAR-10		CIFAR-100	
ResNet-18	Accuracy ↑	SAM% ↑	Accuracy ↑	SAM% ↑
SGD	96.23±0.11	170%	79.84±0.45	171%
SAM	**96.79±0.03**	100%	**81.04±0.29**	100%
LookSAM-5	96.47±0.13	115%	80.48±0.24	113%
ESAM	96.58±0.13	104%	80.47±0.31	105%
ESAM[1]	96.56±0.08	140%	80.41±0.10	140%
SAF	96.15±0.11	147%	80.03±0.43	147%
SAF[2]	96.37±0.02	194%	80.06±0.05	192%
MESA[2]	96.24±0.02	168%	79.79±0.09	165%
K-SAM[3]	96.47±0.05	190%	79.00±0.16	189%
G-RST[4]	96.35±0.10	/	80.05±0.18	/
AUSAM-0.4	96.26±0.10	156%	80.14±0.29	155%
AUSAM-0.5	96.52±0.01	132%	80.67±0.05	132%
AUSAM-0.6	96.59±0.14	123%	80.69±0.15	119%
WideResNet-28-10	Accuracy ↑	SAM% ↑	Accuracy ↑	SAM% ↑
SGD	96.91±0.06	195%	82.47±0.30	195%
SAM	97.44±0.04	100%	84.71±0.21	100%
LookSAM-5	97.13±0.04	154%	83.52±0.09	157%
ESAM	97.41±0.07	144%	84.71±0.32	141%
ESAM[1]	97.29±0.11	139%	84.51±0.01	139%
SAF	96.96±0.09	183%	82.57±0.18	182%
SAF[2]	97.08±0.15	198%	83.81±0.04	197%
MESA[2]	97.16±0.23	168%	83.59±0.24	169%
K-SAM[3]	97.41±0.05	217%	84.05±0.19	204%
G-RST[4]	97.32±0.05	/	83.81±0.15	/
AUSAM-0.4	97.17±0.03	204%	84.44±0.31	202%
AUSAM-0.5	97.3±0.06	179%	84.62±0.26	179%
AUSAM-0.6	**97.49±0.05**	152%	**84.73±0.29**	152%
PyramidNet-110	Accuracy ↑	SAM% ↑	Accuracy ↑	SAM% ↑
SGD	97.14±0.08	195%	83.38±0.21	196%
SAM	97.69±0.09	100%	**86.06±0.16**	100%
LookSAM-5	97.22±0.05	121%	83.76±0.45	126%
ESAM	97.59±0.17	115%	85.32±0.03	116%
ESAM[1]	**97.81±0.01**	139%	85.56±0.05	138%
SAF	96.96±0.05	178%	83.66±0.34	181%
SAF[2]	97.34±0.06	202%	84.71±0.01	200%
MESA[2]	97.46±0.09	171%	84.73±0.14	171%
K-SAM[3]	97.62±0.10	195%	84.60±0.22	204%
AUSAM-0.4	97.51±0.02	203%	84.86±0.25	201%
AUSAM-0.5	97.63±0.04	175%	85.68±0.03	176%
AUSAM-0.6	97.72±0.09	151%	85.78±0.06	153%

1 We report the results in (Du et al. 2022a).
2 We report the results in (Du et al. 2022b).
3 We report the results in (Ni et al. 2022).
4 We report the results in (Zhao, Zhang, and Hu 2022).

Table 2: The results of the proposed method and the comparison methods on CIFAR-10 and CIFAR-100 dataset. SAM% indicate the ratio of corresponding method's training speed to SAM's. ↑ means that the larger the reported results are better. The best accuracy is in bold and the second best is underlined.

	ResNet-18		ResNet-50		MobileNetv1	
Methods	Accuracy	SAM%	Accuracy	SAM%	Accuracy	SAM%
SGD	61.88±0.31	198%	65.60±0.51	201%	58.12±0.4	189%
SAM	<u>64.43±0.08</u>	100%	**67.69±0.10**	100%	58.49±0.38	100%
AUSAM-0.5	64.37±0.03	191%	66.60±0.21	195%	<u>58.55±0.23</u>	185%
AUSAM-0.6	**64.49±0.39**	151%	<u>66.65±0.34</u>	166%	**58.94±0.24**	161%

Table 3: The results of SGD, SAM and AUSAM on Tiny-ImageNet. The best accuracy is in bold and the second best is underlined.

	Hea	Sho	Elb	Wri	Hip	Kne	Ank	Mean	SAM%
Adam	96.8	95.8	89.7	84.3	88.7	85.1	81.4	89.3	183%
SAM	97.0	95.7	89.6	84.2	88.4	85.5	81.9	<u>89.4</u>	100%
AUSAM	96.9	95.7	89.4	84.8	89.1	85.7	81.3	**89.5**	144%

Table 4: Results of training SimCC on MPII with Adam,SAM and AUSAM.

Methods	Accuracy	SAM%
SGD	79.79±0.14	177%
SAM	**81.24±0.10**	100%
SGD+Info	79.21±0.24	197%
SAM+Info*	79.68±0.26	137%
SAM+Info	79.57±0.09	140%
SAM+Random	<u>80.56±0.19</u>	120%

Table 5: The Results of using InfoBatch to accelerate SAM.

	ResNet-18		MobileNets	
Methods	Accuracy	SAM%	Accuracy	SAM%
Full prec.	88.72	\	85.81	\
SGD	88.86±0.18	174%	84.04±0.13	153%
SAM	**89.75±0.21**	100%	<u>84.72±0.11</u>	100%
AUSAM	<u>89.63±0.10</u>	130%	**84.91±0.19**	117%

Table 6: Results of QAT with SGD, SAM and AUSAM on the Cifar-10.

Application to Human Pose Estimation

2D Human Pose Estimation (HPE) aims to localize body joints from a single image. We apply SAM and AUSAM to HPE to evaluate their general applicability, using the SimCC method (Li et al. 2022) for validation. The key idea of SimCC is to treat human pose estimation as two classification tasks, one for vertical and one for horizontal coordinates. We use SGD, SAM, and AUSAM as optimizers for the SimCC, and conducted experiments on the MPII dataset (Andriluka et al. 2014). We follow the evaluation procedure in (Li et al. 2022), and conduct experiments with an image resolution of 256×256. In the experiments, both SAM and AUSAM use the Adam optimizer as their base optimizer, with a batch size set to 64. The results are shown in Tab. 4. It can be found that both SAM and AUSAM obtain better performance than Adam. In terms of speed, AUSAM is faster than SAM but slower than Adam. This suggests that AUSAM is applicable across a broad spectrum of scenarios.

Application to Quantization-Aware Training

Neural network quantization reduces computational demands by lowering weight and activation precision, enabling efficient deployment on edge devices without compromising model performance (Jacob et al. 2018; Esser et al. 2020; Wei et al. 2021; Nagel et al. 2022). We employ AUSAM as the optimizer for QAT (Jacob et al. 2018) to demonstrate its broader applicability. We applied the SGD, SAM, and AUSAM algorithms to quantize the parameters of the ResNet-18 and MobileNetV1 models to W4A4 on the CIFAR-10 dataset. The results are presented in Tab. 6. Although the time required for both forward and backward is significantly reduced after model quantization, experimental results show that AUSAM can still accelerate the optimization speed while maintaining performance.

Conclusions

The findings of this study reveal that during the optimization process of SAM, training with samples that have larger gradient norms from each mini-batch effectively preserves the model's generalization capability without significant deterioration. Based on this observation, this paper proposes an asymptotic unbiased sampling strategy aimed at accelerating SAM. Integrating this strategy into SAM enhances optimization speed by approximately 70% while maintaining the model's generalization capability. AUSAM samples a subset of data from the mini-batch based on the estimated average gradient norm before each optimization iteration, significantly reduces the forward and backward time. This is particularly beneficial for large models that require a significant amount of time for forward and backward. Experiments on the CIFAR-10, CIFAR-100, and Tiny-ImageNet datasets show that AUSAM not only matches SAM in model generalization but also significantly surpasses it in optimization speed. Additionally, we applied AUSAM to human pose estimation and model quantization tasks. The results show that AUSAM enhances optimization speed while preserving performance, demonstrating its broad applicability.

Acknowledgements

The work was supported by the National Key Research and Development Program of China (Grant No. 2023YFC3306401). This research was also supported by Zhejiang Provincial Natural Science Foundation of China under Grant No. LD24F020007, Beijing Natural Science Foundation L223024 and L244043, National Natural Science Foundation of China under Grant 61872333, 62076016, "One Thousand Plan" projects in Jiangxi Province Jxsq2023102268.

References

Andriluka, M.; Pishchulin, L.; Gehler, P.; and Schiele, B. 2014. 2D Human Pose Estimation: New Benchmark and State of the Art Analysis. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Andriushchenko, M.; and Flammarion, N. 2022. Towards understanding sharpness-aware minimization. In *Proceedings of the International Conference on Machine Learning (ICML)*, 639–668. PMLR.

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

Becker, M.; Altrock, F.; and Risse, B. 2024. Momentum-SAM: Sharpness Aware Minimization without Computational Overhead. *arXiv preprint arXiv:2401.12033*.

Cha, J.; Chun, S.; Lee, K.; Cho, H.-C.; Park, S.; Lee, Y.; and Park, S. 2021. Swad: Domain generalization by seeking flat minima. *Advances in Neural Information Processing Systems (NeurIPS)*, 34: 22405–22418.

Chrabaszcz, P.; Loshchilov, I.; and Hutter, F. 2017. A downsampled variant of imagenet as an alternative to the cifar datasets. *arXiv preprint arXiv:1707.08819*.

Deng, J.; Pang, J.; Zhang, B.; and Wang, T. 2024. Effective Gradient Sample Size via Variation Estimation for Accelerating Sharpness aware Minimization. *arXiv preprint arXiv:2403.08821*.

DeVries, T.; and Taylor, G. W. 2017. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*.

Du, J.; Yan, H.; Feng, J.; Zhou, J. T.; Zhen, L.; Goh, R. S. M.; and Tan, V. Y. 2022a. Efficient sharpness-aware minimization for improved training of neural networks.

Du, J.; Zhou, D.; Feng, J.; Tan, V.; and Zhou, J. T. 2022b. Sharpness-aware training for free. *Advances in Neural Information Processing Systems (NeurIPS)*, 35: 23439–23451.

Esser, S. K.; McKinstry, J. L.; Bablani, D.; Appuswamy, R.; and Modha, D. S. 2020. Learned Step Size Quantization. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Foret, P.; Kleiner, A.; Mobahi, H.; and Neyshabur, B. 2021. Sharpness-aware minimization for efficiently improving generalization.

Goodfellow, I. 2015. Efficient per-example gradient computations. *arXiv preprint arXiv:1510.01799*.

Han, D.; Kim, J.; and Kim, J. 2017. Deep pyramidal residual networks. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 5927–5935.

He, F.; Liu, T.; and Tao, D. 2019. Control batch size and learning rate to generalize well: Theoretical and empirical evidence. *Advances in neural information processing systems*, 32.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 770–778.

Hochreiter, S.; and Schmidhuber, J. 1994. Simplifying neural nets by discovering flat minima. *Advances in Neural Information Processing Systems (NeurIPS)*, 7.

Howard, A. G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; and Adam, H. 2017. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*.

Izmailov, P.; Podoprikhin, D.; Garipov, T.; Vetrov, D.; and Wilson, A. G. 2018. Averaging weights leads to wider optima and better generalization. 876–885.

Jacob, B.; Kligys, S.; Chen, B.; Zhu, M.; Tang, M.; Howard, A.; Adam, H.; and Kalenichenko, D. 2018. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2704–2713.

Jiang, W.; Yang, H.; Zhang, Y.; and Kwok, J. 2023. An Adaptive Policy to Employ Sharpness-Aware Minimization. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Jiang, Y.; Neyshabur, B.; Mobahi, H.; Krishnan, D.; and Bengio, S. 2020. Fantastic generalization measures and where to find them.

Keskar, N. S.; Mudigere, D.; Nocedal, J.; Smelyanskiy, M.; and Tang, P. T. P. 2017. On large-batch training for deep learning: Generalization gap and sharp minima. *Proceedings of the International Conference on Learning Representations (ICLR)*.

Krizhevsky, A.; Hinton, G.; et al. 2009. Learning multiple layers of features from tiny images.

Kwon, J.; Kim, J.; Park, H.; and Choi, I. K. 2021. Asam: Adaptive sharpness-aware minimization for scale-invariant learning of deep neural networks. In *Proceedings of the International Conference on Machine Learning (ICML)*, 5905–5914. PMLR.

Le, Y.; and Yang, X. 2015. Tiny imagenet visual recognition challenge. *CS 231N*, 7(7): 3.

Li, Y.; Yang, S.; Liu, P.; Zhang, S.; Wang, Y.; Wang, Z.; Yang, W.; and Xia, S.-T. 2022. Simcc: A simple coordinate classification perspective for human pose estimation. In *European Conference on Computer Vision*, 89–106. Springer.

Liu, Y.; Mai, S.; Chen, X.; Hsieh, C.-J.; and You, Y. 2022. Towards efficient and scalable sharpness-aware minimization. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 12360–12370.

Loshchilov, I.; and Hutter, F. 2016. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*.

Nagel, M.; Fournarakis, M.; Bondarenko, Y.; and Blankevoort, T. 2022. Overcoming oscillations in quantization-aware training. In *Proceedings of the International Conference on Machine Learning (ICML)*, 16318–16330. PMLR.

Neyshabur, B.; Bhojanapalli, S.; McAllester, D.; and Srebro, N. 2017. Exploring generalization in deep learning. *Advances in Neural Information Processing Systems (NeurIPS)*, 30.

Ni, R.; Chiang, P.-y.; Geiping, J.; Goldblum, M.; Wilson, A. G.; and Goldstein, T. 2022. K-sam: Sharpness-aware minimization at the speed of sgd. *arXiv preprint arXiv:2210.12864*.

Paul, M.; Ganguli, S.; and Dziugaite, G. K. 2021. Deep learning on a data diet: Finding important examples early in training. *Advances in Neural Information Processing Systems*, 34: 20596–20607.

Qin, Z.; Wang, K.; Zheng, Z.; Gu, J.; Peng, X.; Zhou, D.; Shang, L.; Sun, B.; Xie, X.; You, Y.; et al. 2024. InfoBatch: Lossless Training Speed Up by Unbiased Dynamic Data Pruning. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Raju, R. S.; Daruwalla, K.; and Lipasti, M. 2021. Accelerating deep learning with dynamic data pruning. *arXiv preprint arXiv:2111.12621*.

Rochette, G.; Manoel, A.; and Tramel, E. W. 2019. Efficient per-example gradient computations in convolutional neural networks. *arXiv preprint arXiv:1912.06015*.

Toneva, M.; Sordoni, A.; Combes, R. T. d.; Trischler, A.; Bengio, Y.; and Gordon, G. J. 2018. An empirical study of example forgetting during deep neural network learning. *arXiv preprint arXiv:1812.05159*.

Wei, X.; Gong, R.; Li, Y.; Liu, X.; and Yu, F. 2021. QDrop: Randomly Dropping Quantization for Extremely Low-bit Post-Training Quantization. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Wu, D.; Xia, S.-T.; and Wang, Y. 2020. Adversarial weight perturbation helps robust generalization. *Advances in Neural Information Processing Systems (NeurIPS)*, 33: 2958–2969.

Zagoruyko, S.; and Komodakis, N. 2016. Wide residual networks. *arXiv preprint arXiv:1605.07146*.

Zhang, C.; Bengio, S.; Hardt, M.; Recht, B.; and Vinyals, O. 2021. Understanding deep learning (still) requires rethinking generalization. *Communications of the ACM*, 64(3): 107–115.

Zhang, X.; Xu, R.; Yu, H.; Dong, Y.; Tian, P.; and Cui, P. 2023. Flatness-Aware Minimization for Domain Generalization. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 5189–5202.

Zhao, Y.; Zhang, H.; and Hu, X. 2022. Randomized sharpness-aware training for boosting computational efficiency in deep learning. *arXiv preprint arXiv:2203.09962*.

Zhuang, J.; Gong, B.; Yuan, L.; Cui, Y.; Adam, H.; Dvornek, N.; Tatikonda, S.; Duncan, J.; and Liu, T. 2022. Surrogate gap minimization improves sharpness-aware training.