Research Article

DGSAM: Domain Generalization via Individual Sharpness-Aware Minimization

Youngjun Song[1], Youngsik Hwang[2], Jonghun Lee[2], Heechang Lee[1], Dong-Young Lim[1,2]

1. Department of Industrial Engineering, Ulsan National Institute of Science and Technology, Ulsan, South Korea; 2. Artificial Intelligence Graduate School, Ulsan National Institute of Science and Technology, Ulsan, South Korea

Domain generalization (DG) aims to learn models that can generalize well to unseen domains by training only on a set of source domains. Sharpness-Aware Minimization (SAM) has been a popular approach for this, aiming to find flat minima in the total loss landscape. However, we show that minimizing the total loss sharpness does not guarantee sharpness across individual domains. In particular, SAM can converge to fake flat minima, where the total loss may exhibit flat minima, but sharp minima are present in individual domains. Moreover, the current perturbation update in gradient ascent steps is ineffective in directly updating the sharpness of individual domains. Motivated by these findings, we introduce a novel DG algorithm, Decreased-overhead Gradual Sharpness-Aware Minimization (DGSAM), that applies gradual domain-wise perturbation to reduce sharpness consistently across domains while maintaining computational efficiency. Our experiments demonstrate that DGSAM outperforms state-of-the-art DG methods, achieving improved robustness to domain shifts and better performance across various benchmarks, while reducing computational overhead compared to SAM.

Youngjun Song and Youngsik Hwang equally contributed to this work.

Corresponding author: Dong-Young Lim, dlim@unist.ac.kr

1. Introduction

The remarkable empirical performance of deep neural networks is largely based on the strong assumption of independent and identically distributed (i.i.d.) data[1]. However, this assumption is often

unrealistic in many real-world applications, highlighting the need for models that are robust under distribution shifts beyond the training data distribution. For example, in medical image classification, the test dataset may differ significantly from training data due to factors such as imaging protocols and device vendor[2]. In object detection for self-driving cars, real-world environments frequently vary from training conditions due to weather and camera settings[3]. However, it is impractical to include every possible scenario in the training data. These primary challenges, also known as *domain shift*, highlight the importance of developing models that can generalize well to unseen domain shifts.

A common approach to address domain shift involves learning domain-invariant features by aligning the distributions of source domains and minimizing their discrepancies[4][5]. Also, methods such as adversarial training[6][7] and data augmentation[8][9][10] have been widely explored to ensure that the learned representations are less sensitive to variations in data-specific variations. More recently, meta-learning strategies[11][12] have tacked domain generalization as a meta-learning problem, simulating domain shifts during training to improve model robustness.

Another line of research focuses on seeking flat minima in the loss landscape, as flatter minima are believed to improve generalization and robustness to distributional shifts[13][14][15][16][17]. A prominent approach in this field is Sharpness-Aware Minimization (SAM)[18], aiming to improve generalization by minimizing both empirical risk and sharpness of the loss surface. SAM perturbs the model parameters in the direction of greatest sharpness to identify flatter regions in the loss landscape. This approach promotes solutions that are less sensitive to variations in input distributions. The principle of SAM[19][20][21] have been widely applied in domain generalization, yielding meaningful performance improvements. However, the relationship between flat minima and robustness to domain shifts remains relatively understudied.

In this paper, we find that SAM-based algorithms for domain generalization may overlook the limitation that minimizing the sharpness of source domains does not necessarily lead to reduced sharpness in individual domains. This hinders SAM from learning domain-invariant features, which can eventually lead to poor generalization on unseen domains. Our analysis provides theoretical support for this issue, and we further validate it through empirical observation. Furthermore, we demonstrate that the current parameter perturbation of SAM increases the total loss, but has a relatively limited impact on individual domain losses. In addition to this, it is necessary to align the eigenvector of the Hessian and perturbation directions, so we constructed the ideal perturbation using second order terms. However, this ideal

strategy is intractable due to the computational cost. To address this, we introduce a new adaptive perturbation strategy which has the same effect, *gradual perturbation*, which aims to find a perturbed parameter that is sensitive to individual domains as well. We confirm that gradual perturbation provides an effective strategy for calculating the perturbed parameter for both source domains and unseen domains.

Based on these observations, we propose a novel DG algorithm, Decreased-overhead Gradual Sharpness-Aware Minimization (DGSAM), which gradually perturbs the parameters using the loss gradient of each domain and finally updates with aggregated gradients. DGSAM improves upon three key aspects of the existing SAM-based approaches for domain generalization. First, it reduces the sharpness of individual domains instead of the total loss sharpness, allowing the model to better learn domain-invariant features. Second, while traditional SAM-based algorithms incur twice the computational overhead compared to empirical risk minimization, DGSAM significantly improves computational efficiency by reusing gradients calculated during the adaptive gradual perturbation. Third, whereas previous methods relied on proxy measures of curvature to achieve flatness, DGSAM directly controls the Hessian's eigenvalues, the most direct measure of curvature[22][23]. Our experimental results show that DGSAM outperforms existing DG algorithms in the DomainBed[24] protocol. Moreover, DGSAM consistently shows high average accuracy and low standard deviation across various datasets, demonstrating its robustness to domain shift s. Notably, DGSAM significantly reduces the sharpness across individual source domains compared to existing SAM-based algorithms, including SAM and SAGM[19].

2. Preliminaries and Related Work

2.1. Domain Generalization

Let $\mathcal{D}_s := \{\mathcal{D}_i\}_{i=1}^{S}$ denote the collection of training samples from different domain sources where $\mathcal{D}_i$ represents the training samples from the i-th domain. We define the total loss function over $\mathcal{D}$ as follows:

$$\mathcal{L}_{\mathcal{D}_s}(\theta) := \frac{1}{|\mathcal{D}_s|} \sum_{\mathcal{D}_i \in \mathcal{D}_s} \mathcal{L}_{\mathcal{D}_i}(\theta) \tag{1}$$

where $\mathcal{L}_{\mathcal{D}_i}$ is the loss function evaluated using the training samples of the i-th domain and θ is the parameter of a given model.

Given a set of source domain samples $\mathcal{D}_s$, the model parameters obtained by naively minimizing the population risk over the source domains, i.e., $\theta_s^* = \text{argmin}_\theta \mathcal{L}_{\mathcal{D}_s}(\theta)$, tend to struggle in generalizing to unseen domain distributions as they are optimized exclusively on the source domains. Therefore, the primary goal of domain generalization is to learn model parameters θ that are robust to domain shifts and generalize well to unseen domains when trained solely on source domains.

As the importance of Domain Generalization (DG) has grown, various datasets[25][26][27] and benchmark sets[24][28] have been developed to evaluate DG methods. Research directions in Domain Generalization (DG) include domain-adversarial learning[29][6][30][31][32], minimizing moments[33][4][34], and contrastive loss[35][36] for domain alignment to create domain-agnostic models. Other approaches focus on data augmentation[37][38][39], domain disentanglement[40][41], meta learning[42][43][11], and ensemble learning[13][44][45].

2.2. Sharpness-Aware Minimization

The relationship between the curvature of loss landscape and model generalization ability has been extensively studied in the literature[46][47][48][17][18]. Motivated by this insight,[18] proposed Sharpness-Aware Minimization (SAM), an optimization framework that enhances generalization by simultaneously minimizing an associated loss function $\mathcal{L}(\theta)$ and penalizing sharpness. The objective of SAM is to minimize a perturbed loss $\mathcal{L}^p(\theta)$ as follows:

$$\min_\theta \mathcal{L}^p(\theta) = \min_\theta \max_{\|\epsilon\|_2 \leq \rho} \mathcal{L}(\theta + \epsilon)$$

where $\rho > 0$ represents the radius of the perturbation ϵ. In practice, the solution ϵ^* of the inner maximization is approximated using a first-order Taylor expansion and the dual norm formulation:

$$\epsilon^* = \text{argmax}_{\|\epsilon\|_2 \leq \rho} \mathcal{L}(\theta + \epsilon) \approx \text{argmax}_{\|\epsilon\|_2 \leq \rho} \epsilon^\top \nabla \mathcal{L}(\theta) = \rho \frac{\nabla \mathcal{L}(\theta)}{\|\nabla \mathcal{L}(\theta)\|_2}.$$

Then, the objective of SAM reduces to

$$\min_\theta \mathcal{L}(\theta + \epsilon^*).$$

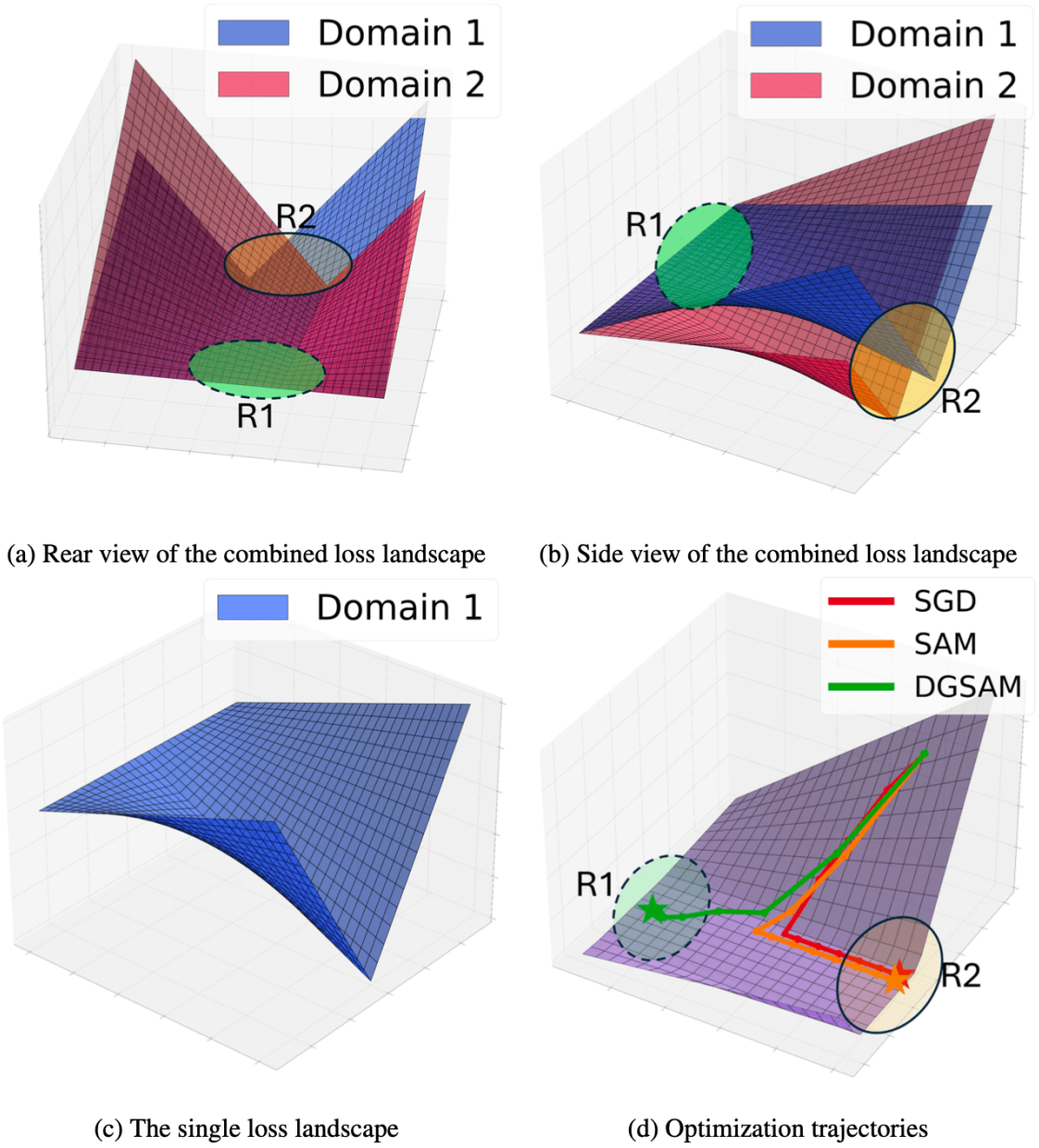
Following the SAM, several studies have focused on finding flat minima. ASAM[49] defined adaptive sharpness, which modifies ρ adaptively, and GSAM[50] introduced a surrogate gap $\mathcal{L}^p(\theta) - \mathcal{L}(\theta)$ that better agrees with sharpness as opposed to merely reducing the perturbed loss. GAM[51] introduced first-order flatness, which represents the curvature of the loss landscape, to minimize the sensitivity of the

landscape more explicitly. Additionally, Lookahead optimizer and Lookbehind-SAM[52][53] modified the two-step structure to perform multiple steps per iteration.

While SAM and its variants[18][50][19] have demonstrated significant improvements in generalization, a major drawback is their computational overhead. Specifically, these methods require performing backpropagation twice in each iteration: first to calculate the perturbation direction and then to update the model parameters, leading to a computational cost that is double that of ERM. ESAM and LookSAM[54][55] were introduced to mitigate computational overhead while preserving SAM's performance.

In domain generalization,[19][20][13][56] have utilized sharpness-aware learning to find flatter minima by reducing the sharpness of the total loss across source domains. Some approaches that integrate domain information into SAM, such as[21] and[57], either focus on the loss variance or apply SAGM on a domain-by-domain basis.

3. Motivation



(a) Rear view of the combined loss landscape

(b) Side view of the combined loss landscape

(c) The single loss landscape

(d) Optimization trajectories

Figure 1. Toy example: two conflicting loss functions construct two different type of flat minima. An interactive visualization of toy example is available at https://dgsam-toy-example.netlify.app/.

Recent studies[14][15] have demonstrated that domain distribution shifts can be viewed as parameter perturbations. Specifically, for two given domain samples $\mathcal{D}_i$, $\mathcal{D}_j$ and a model parameter θ, there exists a parameter perturbation v such that $\mathcal{L}_{\mathcal{D}_i}(\theta) = \mathcal{L}_{\mathcal{D}_j}(\theta + v)$. This finding indicates that minimizing the

doi.org/10.32388/CLU2QF

perturbed loss is closely connected to robustness to domain shifts, providing theoretical justification for the use of SAM in DG.

Inspired by this theoretical support along with the strong generalization ability of SAM, the concept of SAM has been widely employed in DG[19][56][57][20]. Recall $\mathcal{D}_s$ is the set of S source domain training samples. Then, SAM for DG considers the following optimization problem:

$$\min_{\theta} \max_{\|\epsilon\|_2 \leq \rho} \mathcal{L}_{\mathcal{D}_s}(\theta + \epsilon) \tag{3.1}$$

where $\mathcal{L}_{\mathcal{D}_s}(\cdot)$, defined in Eq. (2.1), is the total loss function over $\mathcal{D}_s$. Let

$$\mathcal{S}_{\mathcal{D}_s}(\theta) = \max_{\|\epsilon\|_2 \leq \rho} \mathcal{L}_{\mathcal{D}_s}(\theta + \epsilon) - \mathcal{L}_{\mathcal{D}_s}(\theta)$$

denote the zeroth-order sharpness of the total loss. Then, the objective of SAM for DG can be rewritten as

$$\min_{\theta} \mathcal{S}_{\mathcal{D}_s}(\theta) + \mathcal{L}_{\mathcal{D}_s}(\theta).$$

In other words, a straightforward implementation of SAM for DG aims to minimize the total loss and its zeroth-order sharpness.

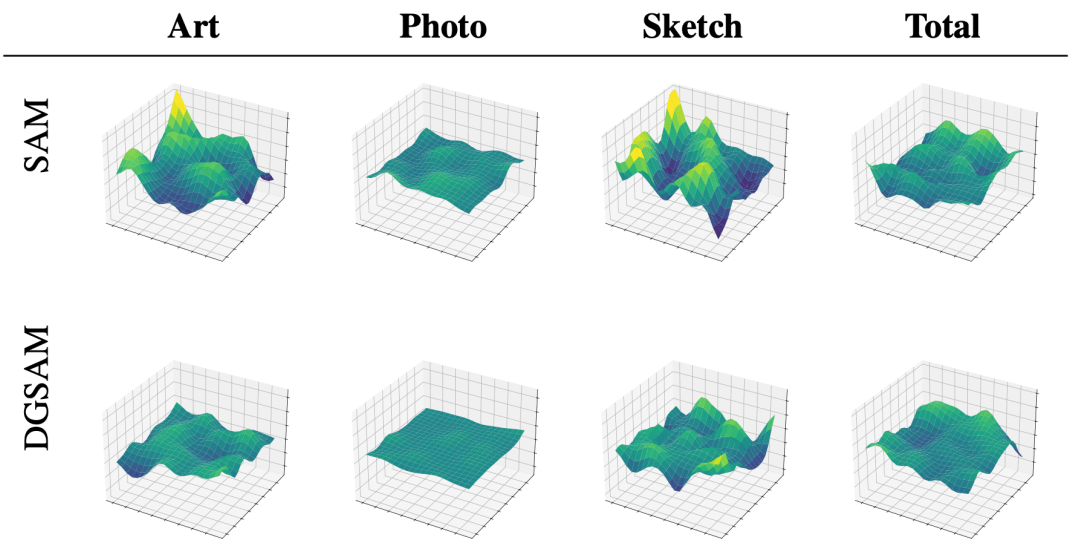


Figure 2. Comparison of loss landscapes of converged minima using SAM and DGSAM across different domains on the PACS dataset. We set the grid with two random direction. DGSAM performs better than SAM in reducing individual sharpness in all three individual domains, and total sharpness.

doi.org/10.32388/CLU2QF

To generalize to unseen domains using only source domains, DG requires the model to avoid overfitting to the idiosyncratic features of each source domain. Instead, it should focus on generalizing to the shared features between unseen and source domains. Therefore, achieving flat minima at the individual domain level is essential. The rationale for applying SAM and its variants in DG is based on the idea that decreasing the sharpness of the total loss will potentially reduce the sharpness for each individual domain loss, eventually leading to robust performance on unseen domains associated with each source domain. However, Proposition 3.1 reveals that the sharpness of the total loss does not necessarily reduce the average sharpness across individual domains.

Proposition 3.1. *Consider the total loss function* $\mathcal{L}_{\mathcal{D}_s}(\theta) = \frac{1}{S}\sum_{i=1}^{S}\mathcal{L}_{\mathcal{D}_i}(\theta)$*, where* $\mathcal{L}_{\mathcal{D}_i}$ *is the individual loss function. Let* $\mathcal{S}_{\mathcal{D}_s}(\theta)$ *represent the zeroth-order sharpness of the total loss function, and let* $\mathcal{S}_i(\theta)$ *denote the zeroth-order sharpness of the* i*-th loss function* $\mathcal{L}_{\mathcal{D}_i}$*. Then, for two different local minima* θ_1 *and* θ_2*,*

$$\mathcal{S}_{\mathcal{D}_s}(\theta_1) < \mathcal{S}_{\mathcal{D}_s}(\theta_2) \Longrightarrow \frac{1}{S}\sum_{i=1}^{S}\mathcal{S}_i(\theta_1) < \frac{1}{S}\sum_{i=1}^{S}\mathcal{S}_i(\theta_2).$$

We refer the supplement for the proof of Proposition 3.1. Proposition 3.1 implies that a careless adoption of SAM in DG may fail to achieve flat minima at the individual domain level.

To illustrate this phenomenon, we present a toy example that considers a 2-dimensional minimization problem involving two loss functions. Note that each loss function corresponds to the loss function from different domain. The two loss functions share the same loss landscape (see Figure 1(c)), but one is obtained by shifting the other along one axis. Figure 1(a) and 1(b) show the two loss functions from different angles. Both loss functions have relatively flat minima in the region **R1** shaded in green, while the region **R2** shaded in yellow indicates sharp minima. However, when considering the sum of the two loss functions, both regions show flat minima as shown in Figure 1(d). The reason is that in the region **R2**, the two sharp valleys can create a flat region when combined as illustrated in Figure 3. Thus, the region **R1** where two domain losses have flat minima, represents the ideal solution. In contrast, although the region **R2** appears flat in the total loss, it should be avoided because both individual domain losses exhibit sharp region there, making it *fake flat minima*.

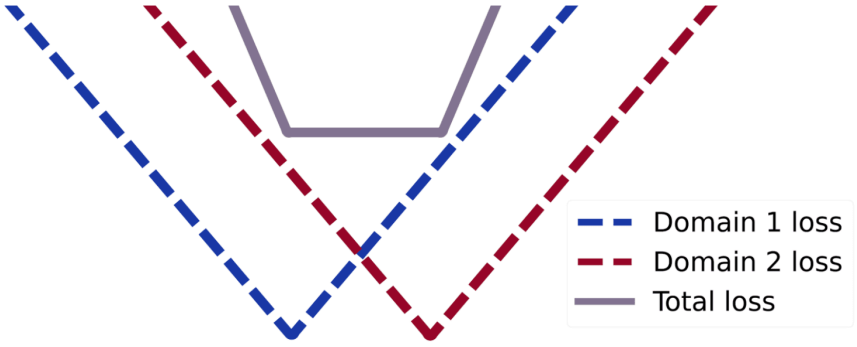


Figure 3. The sum of two sharp losses can result in a flat total loss.

This example shows that minimizing the total loss sharpness does not ensure flat minima for individual losses. In fact, it may lead to sharp minima at the individual domain level. When solved with SAM and SGD, both methods converge to the fake flat minima in the region **R2**.

Beyond this simple toy example, such an issue is consistently observed in practical DG tasks. Figure 2, shows the visualization for loss landscape of converged minima using SAM and DGSAM on ResNet-50. SAM achieves flat minima in the total loss but fails to find flat minima at the individual domain level. These findings suggest the need for a new SAM approach for DG that accounts for the sharpness of each individual loss instead of minimizing total loss sharpness.

4. Methodology

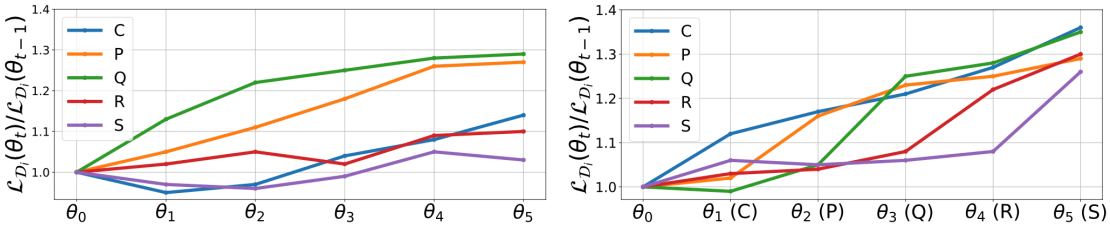
4.1. Failure of Total Gradient Perturbation in Increasing Domain-wise Loss

At each iteration t, SAM performs gradient ascent to find the direction that maximizes the loss where the model is most sensitive by perturbating the parameters as follows:

$$\tilde{\theta}_t = \theta_t + \epsilon^*_{\mathcal{D}_s} = \theta + \rho \frac{\nabla \mathcal{L}_{\mathcal{D}_s}(\theta_t)}{\|\nabla \mathcal{L}_{\mathcal{D}_s}(\theta_t)\|}. \tag{4.1}$$

We note that $\epsilon^*_{\mathcal{D}_s}$ is calculated based on the gradient of the total loss $\nabla \mathcal{L}_{\mathcal{D}_s}(\theta_t)$. However, the perturbation using $\epsilon^*_{\mathcal{D}_s}$ may not yield the optimal perturbed parameter for minimizing individual domain losses, as the total loss gradient does not align with the gradients of individual domain losses, $\nabla \mathcal{L}_{\mathcal{D}_i}(\theta_t)$ for $i = 1, 2, \ldots, S$, as discussed in Section 3.

doi.org/10.32388/CLU2QF



(a) Perturb by total gradient. (b) Perturb by individual gradient.

Figure 4. (a) Perturb by total gradient. (b) Perturb by individual gradient.Loss increment across domains by perturbation at each ascent step.

Figure 4 illustrates the effect of different perturbation directions on domain-wise loss variations. Starting from the initial parameter θ_0, we iteratively apply perturbations ϵ_t to obtain the perturbed parameter $\tilde{\theta}_i = \theta_0 + \sum_{j=1}^{i} \epsilon_j$ for the ResNet-50[58] model on the DomainNet[27] dataset. In Figure 4(a), the perturbation direction is given by the total gradient as $\epsilon_i = \rho \frac{\nabla \mathcal{L}_{\mathcal{D}_s}(\tilde{\theta}_{i-1})}{\|\nabla \mathcal{L}_{\mathcal{D}_s}(\tilde{\theta}_{i-1})\|}$. On the other hand, in Figure 4(b), perturbations are applied sequentially using individual domain gradients as $\epsilon_i = \rho \frac{\nabla \mathcal{L}_{\mathcal{D}_{it}}(\tilde{\theta}_{i-1})}{\|\nabla \mathcal{L}_{\mathcal{D}_{it}}(\tilde{\theta}_{i-1})\|}$.

Figure 4(a) shows that perturbing along the total gradient direction results in an imbalanced increase in domain losses, with some domains exhibiting substantial growth while others change minimally. In contrast, Figure 4(b) demonstrates that sequential perturbations based on individual domain gradients produce a more uniform increase in losses across domains. This observation highlights that sequentially perturbing along domain-specific gradients better aligns with the goal of reducing individual sharpness, which is crucial for improving robustness to domain shifts.

4.2. Decreased-overhead Gradual SAM

Based on these observations, we propose a novel domain generalization algorithm, *Decreased-overhead Gradual Sharpness-Aware Minimization* (DGSAM). In the gradient ascent step to find the optimal perturbed parameter, DGSAM uses a gradual strategy: perturbations are applied iteratively S times, each using the optimal perturbation calculated for an individual domain (see lines 7-9 in Algorithm 1). During this process, the gradients for each individual domain loss are stored and later reused during the descent step to improve computational efficiency. However, the gradient for the initial perturbation is computed based on the current parameter θ_t rather than the perturbed parameter. Therefore, an extra perturbation is

needed on the main used for the first calculation to compute the correct domain loss gradient (see lines 10-11 in Algorithm 1).

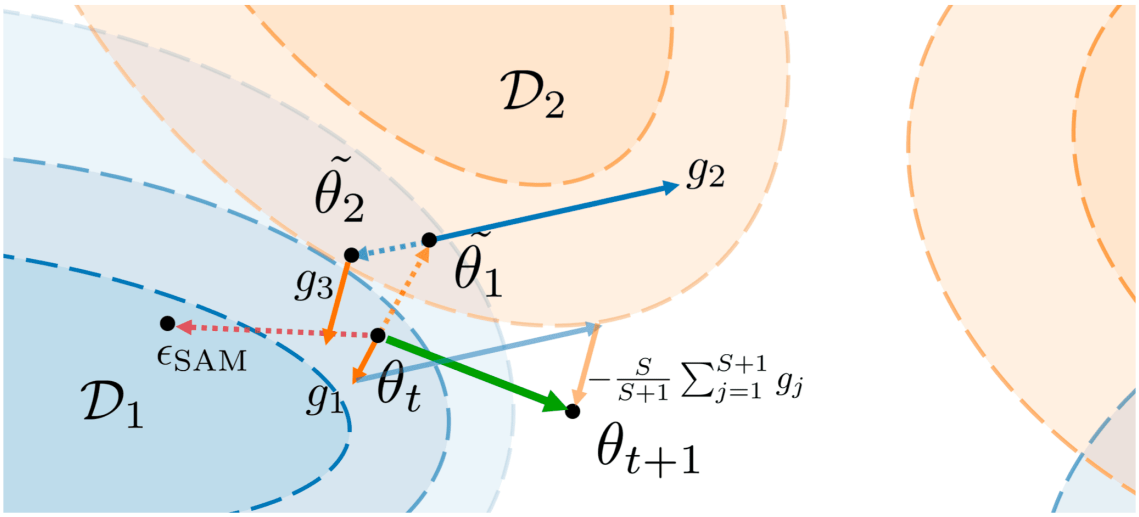


Figure 5. A visualization of DGSAM algorithm.

As a result, DGSAM obtains a perturbed parameter which takes into account the sharpness of each individual domain and collects the gradients for all S domains through $S+1$ computations. Then, the model is updated using the average of these individual domain gradients (see line 14 in Algorithm 1). Figure 5 provides a visualization of our algorithm. When the losses of the two domains overlap, the perturbation direction of SAM is biased toward $\mathcal{D}_1$. In contrast, the parameters $\tilde{\theta}_1, \tilde{\theta}_2$ updated by DGSAM move in a direction that increases loss in both domains, and the subsequent descent to obtain θ_{t+1} reduces sharpness across both domains. This mechanism offers an intuitive explanation for the phenomena observed in Figure 4. In addition, DGSAM requires $S+1$ computations per iteration, which is significantly lower than the $2S$ computations needed by SAM.

doi.org/10.32388/CLU2QF

Algorithm 1 DGSAM

Require: Initial parameter θ_0, learning rate γ, batch size, dropout rate, and weight decay; radius ρ; total iterations N; training sets from S domains $\{\mathcal{D}_i\}_{i=1}^{S}$
for $t \leftarrow 0$ to $N-1$ **do**
  Sample batches $B_i \sim \mathcal{D}_i$ for $i = 1, \cdots, S$
  Set a random order $l = permute(\{1, \cdots, S\})$
  $\widetilde{\theta}_0 \leftarrow \theta_t$
  for $j \leftarrow 1$ to $S+1$ **do**
    if $j \leq S$ **then**
      $g_j \leftarrow \nabla \mathcal{L}_{B_{l_j}}(\widetilde{\theta}_{j-1})$
      $\widetilde{\theta}_j \leftarrow \widetilde{\theta}_{j-1} + \rho \frac{g_j}{\|g_j\|}$
    else if $j = S+1$ **then**
      $g_j \leftarrow \nabla \mathcal{L}_{B_{l_1}}(\widetilde{\theta}_{j-1})$
    end if
  end for
  $\theta_{t+1} \leftarrow \theta_t - \gamma \left(\frac{S}{S+1} \right) \sum_{j=1}^{S+1} g_j$
end for

4.3. Sharpness-Awareness of DGSAM for Individual Domains

Recently work[59][50] has shown that SAM's nested approximations can be problematic, highlighting the need for more direct control over eigenvalues. [60] demonstrated that aligning the perturbation direction with an eigenvector can control the corresponding eigenvalue. However, relying solely on the top eigenvectors falls short in multi-domain scenarios with conflicting gradients. Therefore, it is preferable to identify eigenvectors associated with large eigenvalues and determine a common direction among them across all domains. Moreover, [61] showed that controlling the overall eigenvalue spectrum yields a tighter generalization bound than focusing solely on the top eigenvalue.

In this regard, we provide a detailed analysis to show how the gradual perturbation strategy of DGSAM effectively controls the sharpness of individual domains. In the j-th perturbation step, the gradient g_j is given by:

$$\begin{aligned} g_j = \nabla \mathcal{L}_{B_{l_j}}(\tilde{\theta}_{j-1}) &= \nabla \mathcal{L}_{B_{l_j}} \left(\tilde{\theta}_0 + \sum_{k=1}^{j-1} \rho \frac{g_k}{\|g_k\|} \right) \\ &= \nabla \mathcal{L}_{B_{l_j}}(\tilde{\theta}_0) + \rho \nabla^2 \mathcal{L}_{B_{l_j}}(\tilde{\theta}_0) \sum_{k=1}^{j-1} \frac{g_k}{\|g_k\|} + O(\rho^2) \end{aligned}$$

where B_{l_j} is the minibatch from jth chosen domain. Since any Hessian matrix is diagonalizable, we have $\nabla^2 \mathcal{L}_{B_{l_j}}(\tilde{\theta}_0) = \sum_n \lambda_n v_n v_n^\top$ where $E_j = \{(\lambda_n v_n)\}$ is the set of eigenpairs of $\nabla^2 \mathcal{L}_{B_{l_j}}(\theta_t)$. Then, the g_j can be approximated as

$$g_j \approx \nabla \mathcal{L}_{B_{l_j}}(\tilde{\theta}_0) + \rho \sum_{(\lambda,v)\in E_j} \lambda \left(\sum_{k=1}^{j-1} \frac{v^\top g_k}{\|v\|\|g_k\|} \right) v. \tag{4.2}$$

In this approximation, the first term represents the standard ascent direction for the j-th domain, while the second term is a weighted sum of eigenvectors. The weights reflect both the corresponding eigenvalues and the similarity between the ascent directions from different domains. Thus, the gradual perturbation strategy of DGSAM effectively leverages eigenvector information across all domains, ensuring that the sharpness of individual domain losses is balanced and robustly controlled.

In Figure 10 of the supplement, which compares the magnitudes of the first and second terms, we observe that the second term is of significant magnitude relative to the first term, indicating that incorporating the eigenvalue-weighted eigenvector component substantially alters the vanilla ascent direction. Moreover, in the toy example discussed in Section 3, DGSAM converges to a flat region across all individual domains, thereby avoiding the fake flat minima.

Algorithm	PACS		VLCS		OfficeHome		TerraInc		DomainNet		Avg	
	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD
IRM†[5]	83.5±1.0	8.4	78.6±0.6	12.4	64.3±2.3	9.1	47.6±1.4	7.9	33.9±2.9	15.2	61.6	10.6
ARM†[43]	85.1±0.6	8.0	77.6±0.7	13.1	64.8±0.4	10.2	45.5±1.3	7.4	35.5±0.5	16.7	61.7	11.1
VREx†[62]	84.9±1.1	7.6	78.3±0.8	12.4	66.4±0.6	9.9	46.4±2.4	6.9	33.6±3.0	**15.0**	61.9	10.4
CDANN†[63]	82.6±0.9	9.2	77.5±1.0	12.1	65.7±1.4	10.6	45.8±2.7	**5.9**	38.3±0.5	17.3	62.0	11.0
DANN†[7]	83.7±1.1	9.2	78.6±0.6	12.6	65.9±0.7	9.8	46.7±1.6	7.9	38.3±0.4	17.0	62.6	11.3
RSC†[64]	85.2±1.0	7.6	77.1±0.7	13.0	65.5±1.0	10.0	46.6±1.0	7.0	38.9±0.7	17.3	62.7	11.0
MTL†[65]	84.6±1.0	8.0	77.2±0.8	12.5	66.4±0.5	10.0	45.6±2.4	7.3	40.6±0.3	18.4	62.9	11.2
MLDG†[42]	84.9±1.1	7.9	77.2±0.8	12.2	66.8±0.8	9.9	47.8±1.7	7.6	41.2±1.7	18.4	63.6	11.2
ERM†	85.5±0.6	7.0	77.3±1.1	12.5	67.0±0.4	10.5	47.0±1.0	7.6	42.3±0.4	19.1	63.8	11.4
SagNet†[66]	86.3±0.5	6.9	77.8±0.7	12.5	68.1±0.3	9.5	48.6±0.3	7.1	40.3±0.3	17.9	64.2	10.8
CORAL†[67]	86.2±0.6	7.5	78.8±0.7	12.0	68.7±0.4	9.6	47.7±0.4	7.0	41.5±0.3	18.3	64.6	10.9
SWAD[13]	88.1±0.4	5.9	79.1±0.4	12.8	70.6±0.3	9.2	**50.0**±0.3	7.9	**46.5**±0.2	19.9	66.9	11.2
GAM‡[51]	86.1±1.3	7.4	78.5±1.2	12.5	68.2±0.8	12.8	45.2±1.7	9.1	43.8±0.3	20.0	64.4	12.4
SAM†[18]	85.8±1.3	6.9	79.4±0.6	12.5	69.6±0.3	9.5	43.3±0.3	7.5	44.3±0.2	19.4	64.5	11.2
Lookbehind-SAM[53]	86.0±0.4	7.2	78.9±0.8	12.4	69.2±0.6	11.2	44.5±1.0	8.2	44.2±0.3	19.6	64.7	11.8

Algorithm	PACS		VLCS		OfficeHome		TerraInc		DomainNet		Avg	
	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD
GSAM†[50]	85.9±0.3	7.4	79.1±0.3	12.3	69.3±0.1	9.9	47.0±0.1	8.8	44.6±0.3	19.8	65.2	11.6
FAD[56]	88.2±0.6	6.3	78.9±0.9	12.1	69.2±0.7	13.4	45.7±1.6	9.6	44.4±0.3	19.5	65.3	12.2
DISAM[21]	87.1±0.5	5.6	79.9±0.2	12.3	70.3±0.2	10.3	46.6±1.4	6.9	45.4±0.3	19.5	65.9	10.9
SAGM[19]	86.6±0.3	7.2	80.0±0.4	12.3	70.1±0.3	9.4	48.8±0.3	7.5	45.0±0.2	19.8	66.1	11.2
DGSAM	**88.5**±0.4	**5.2**	**81.4**±0.5	**11.5**	**70.8**±0.3	**8.5**	49.9±0.7	6.9	45.5±0.3	19.4	**67.2**	**10.3**
DGSAM + SWAD	88.7±0.4	5.4	80.9±0.5	11.6	71.4±0.4	8.7	51.1±0.8	6.8	47.1±0.3	19.6	67.8	10.4

Table 1. We compared the performance of DGSAM with 20 baseline algorithms on DomainBed's five datasets. The specific experimental results for each dataset are attached in the supplement. The table presents two types of standard deviation (SD) values. One represents the trial-based SD, calculated across each trial and denoted by the ± symbol adjacent to the mean. The other corresponds to the test domain-specific SD, derived across different test domains and reported separately. Higher Mean means better, and lower SD means better. The best performance except DGSAM + SWAD is highlighted in bold and the second best in underlined. The outcomes of the experiments were marked as † if sourced from[19], ‡ if sourced from[56], and if unlabeled, the data were sourced from individual papers.

5. Numerical Experiments

5.1. Experimental Settings

Evaluation protocols, Baselines and Datasets For all main experiments, we adhere to the DomainBed protocol[24], including model initialization, hyperparameter tuning, and validation methods, to ensure a fair comparison. We evaluate our algorithm across five benchmark datasets widely used in the literature on domain generalization[21][19][13], including PACS[25], VLCS[26], OfficeHome[68], TerraIncognita[69], and DomainNet[27].

We employed leave-one-out cross-validation, a method proposed by[24]. This involves training on all source domains except one target domain and then selecting a model based on its performance on the validation set of the source domains to evaluate accuracy on the target domain. In addition to the original DomainBed protocol, which only reports the average of the performance over each test domain, we also report the standard deviation of the performance varying test domain. This standard deviation serves as a metric of how robust the performance is to the choice of test domain and is used to evaluate the domain-agnostic robustness of our algorithm. To ensure the reliability of our results, we repeated each experiment three times, and the standard errors of these results are included in the supplement.

Implementation Details We used a ResNet-50[58] backbone pretrained on ImageNet, and Adam[70] as the base optimizer. We used the hyperparameter space, the total number of iterations, and checkpoint frequency based on[19]. The specific hyperparameter space and optimal settings for replication are described in the supplement.

5.2. Main Experimental Results

DGSAM outpeforms all baselines on three datasets - PACS, VLCS, and OfficeHome - and achieves comparable performance with SWAD on the remaining two datasets. It is worth noting that a direct comparison between DGSAM (a single optimizer) and SWAD (an ensemble from single trajectory method) is not entirely fair. However, we include SWAD as a SOTA baseline for completeness. Nevertheless, DGSAM not only outperforms SWAD in several cases but also achieves at least comparable results. Furthermore, DGSAM operates on a distinct mechanism from SWAD, making their combination complementary. This synergy enhances performance on DG task.

5.2.1. Variance of Domain-wise Performance

A comprehensive assessment of domain generalization should take into account both the average performance across domains and the variance in performance. When a specific domain is held out for the test domain, its performance is highly dependent on its similarity to the source domains. An ideal robust domain generalization method should exhibit consistent performance across a variety of distribution shifts, ensuring uniform per-domain results regardless of the train-test domain combinations.

Relying solely on the average performance across domains, widely used in DG tasks[24][19][13], can be misleading. A high average may be driven by exceptional performance on test domains that exhibit strong similarity to the source domains, thereby masking poor generalization capability on more

dissimilar domains. This can result in an overestimation of the model's true domain generalization capability.

Therefore, we include the variance (or standard deviation) of domain-wise performance along with the average as a key evaluation metric for domain generalization. This provides a more comprehensive and nuanced understanding of a model's robustness to diverse and potentially unforeseen distributional shifts.

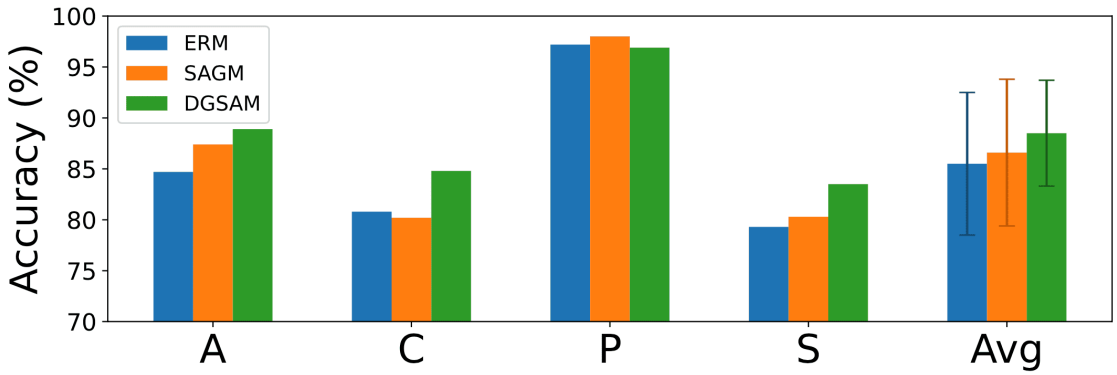


Figure 6. Comparison of accuracy of ERM, SAGM and DGSAM on PACS dataset.

In Figure 6, we compare the per-domain performance of ERM, SAGM, and our proposed method on the PACS dataset. Note that SAGM is the existing SOTA approach that applies SAM to domain generalization. While SAGM achieves a higher average accuracy than ERM, its performance gains are marginal (or even worse) in domains C and S, where ERM performs particularly poorly. In contrast, DGSAM slightly reduces performance on the already high-performing domain P, but significantly improves performance on the other domains. Consequently, DGSAM attains not only a higher accuracy but also a lower variance across domains. This finding emphasizes the importance of including the variance of domain-wise performance as a key evaluation metric and demonstrates that DGSAM learns a more domain-agnostic representation, enhancing its robustness to diverse distributional shifts.

5.2.2. Computational Cost

Our proposed method not only outperforms other algorithms, but also effectively reduces the excessive computational cost commonly associated with SAM variants. Suppose that there are S source domains and the cost of processing a mini-batch from a single domain with ERM is c. Then, the total cost per

iteration for ERM is $S \times c$. In contrast, SAM requires two backpropagation passes for the entire batch of S domains, resulting in a cost of approximately $2S \times c$. DGSAM, on the other hand, computes gradients $S + 1$ times per iteration, yielding a total cost of $(S + 1) \times c$ (see Figure 7).

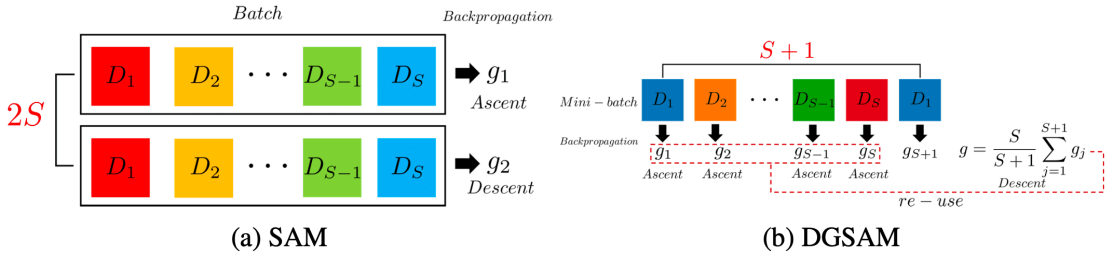


Figure 7. Computational cost of SAM and DGSAM.

To validate this analysis, we measured the computational costs of ERM, SAM, and DGSAM on the PACS dataset, as illustrated in Figure 8. In this experiment, with $S = 3$ source domains, we found that $c \approx 0.37$. SAM exhibited a cost of 0.217, nearly double that of ERM. In contrast, DGSAM achieved a cost of 0.169, which is slightly higher than the theoretical cost of $(S + 1) \times c \approx 0.147$. This small discrepancy arises from extra operations such as gradient summation. These results demonstrate that our algorithm effectively reduces the computational overhead compared to SAM. A comprehensive efficiency comparison on all five DomainBed datasets is provided in the supplement.

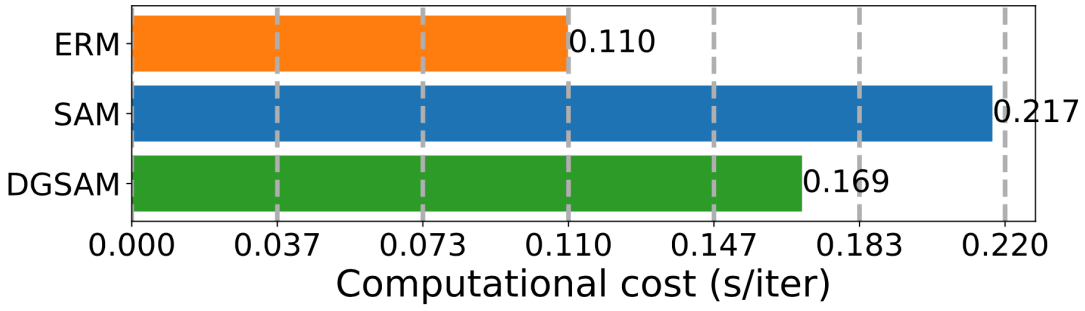


Figure 8. Comparison of empirical computational cost measured by training time per iteration.

doi.org/10.32388/CLU2QF

5.3. Sharpness Analysis

To evaluate whether DGSAM finds flatter minima in individual domains, we compare the sharpness of solutions obtained by DGSAM and SAM. Table 2 shows the zeroth-order sharpness for each domain on the DomainNet dataset. DGSAM consistently achieves lower sharpness in both the source domain losses and the total loss compared to SAM, indicating that it is more effective at finding flatter minima by leveraging domain alignment and direct eigenvalue control. Moreover, DGSAM exhibits significantly lower sharpness in unseen domains, suggesting that reducing sharpness across source domains enhances robustness against domain shifts.

	Individual domains					**Mean (Std)**	**Total**	**Unseen**
	Clipart	**Painting**	**Quickdraw**	**Real**	**Sketch**			
SAM	1.63	6.22	7.86	4.89	3.38	4.79 (2.17)	19.68	70.59
DGSAM	**1.17**	**2.78**	**4.74**	**4.39**	**1.80**	**2.98 (1.40)**	**6.41**	**42.46**

Table 2. The zeroth order sharpness result at converged minima

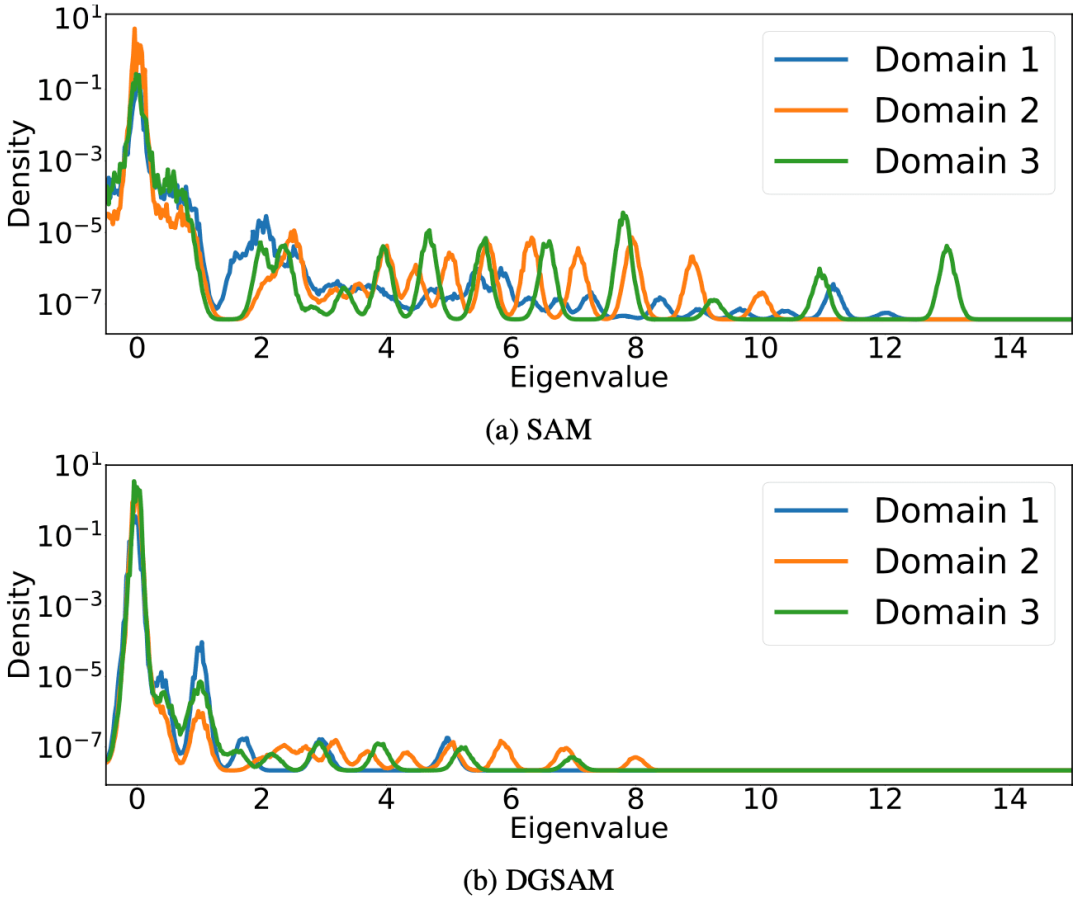


Figure 9. Hessian Spectrum Density at Converged Minima. (a) Eigenvalue distribution per domain for (a) SAM and (b) DGSAM.

We further demonstrate the effectiveness of our approach by estimating the Hessian spectrum density of the converged minima using stochastic Lanczos quadrature[23]. As shown in Figure 9, DGSAM not only suppresses high eigenvalues but also those near zero, indicating an overall control of the eigenvalue spectrum—consistent with our design goals.

Figure 2 visualizes the loss landscape around the solutions for SAM and DGSAM across different domains on the PACS dataset. The loss values are evaluated using random directional perturbations. While the total loss landscape for DGSAM and SAM remains similar, DGSAM finds significantly flatter minima at the individual domain level, whereas SAM converges to fake flat minima.

6. Conclusion and Discussion

In this work, we identified a key limitation of existing SAM-based algorithms: while they reduce the overall loss sharpness, they fail to address the sharpness of individual domains, leading to suboptimal generalization in domain generalization tasks. To overcome this challenge, we introduced Decreased-overhead Gradual Sharpness-Aware Minimization (DGSAM), which sequentially applies perturbations to each domain and aggregates the corresponding gradients. This approach not only facilitates domain alignment but also enables more direct control over large eigenvalues. By reusing gradients computed during the gradual perturbation, DGSAM achieves significantly reduced computational overhead. Extensive experiments demonstrate that DGSAM consistently outperforms current DG methods across various benchmarks while also achieving substantially lower sharpness in individual domains.

While our work offers a promising approach to applying SAM in domain generalization, further investigation is needed to fully establish DGSAM's optimality. For instance, identifying the truly optimal flat minima remains challenging when all local minima are fake flat. Developing an optimizer that consistently converges to the optimal solution would be a valuable extension.

Appendix A. Proof of Proposition 3.1

Proof of Proposition 3.1. Suppose that a local minima θ is given and ρ is sufficiently small. Then, the second-order Taylor expansion for $\mathcal{L}_{\mathcal{D}_s}$ and $\mathcal{L}_{\mathcal{D}_i}$ gives:

$$\mathcal{L}_{\mathcal{D}_s}(\theta+\epsilon) = \mathcal{L}_{\mathcal{D}_s}(\theta) + \nabla\mathcal{L}_{\mathcal{D}_s}(\theta)^\top\epsilon + \frac{1}{2}\epsilon^\top H(\theta)\epsilon + \mathcal{O}(\|\epsilon\|^3)$$

and

$$\mathcal{L}_{\mathcal{D}_i}(\theta+\epsilon) = \mathcal{L}_{\mathcal{D}_i}(\theta) + \nabla\mathcal{L}_{\mathcal{D}_i}(\theta)^\top\epsilon + \frac{1}{2}\epsilon^\top H_i(\theta)\epsilon + \mathcal{O}(\|\epsilon\|^3), \quad i=1,\dots,S$$

where H and H_i are the Hessian matrices for $\mathcal{L}_{\mathcal{D}_s}$ and $\mathcal{L}_{\mathcal{D}_i}$, respectively, evaluated at θ.

Then, using $\nabla\mathcal{L}_{\mathcal{D}_s}(\theta)=0$ and $H(\theta)=\frac{1}{S}\sum_{i=1}^{S}H_i(\theta)$, we have

$$\mathcal{L}_{\mathcal{D}_s}(\theta+\epsilon) - \mathcal{L}_{\mathcal{D}_s}(\theta) = \frac{1}{2}\epsilon^\top\left(\frac{1}{S}\sum_{i=1}^{S}H_i(\theta)\right)\epsilon + \mathcal{O}(\|\epsilon\|^3)$$

which yields the zeroth-order sharpness for $\mathcal{L}_{\mathcal{D}_s}$:

$$\mathcal{S}_{\mathcal{D}_s}(\theta) = \max_{\|\epsilon\|_2\le\rho}\left(\mathcal{L}_{\mathcal{D}_s}(\theta+\epsilon)-\mathcal{L}_{\mathcal{D}_s}(\theta)\right) = \frac{1}{2S}\rho^2\sigma_{\max}\left(\sum_{i=1}^{S}H_i(\theta)\right) + \mathcal{O}(\|\rho\|^3)$$

where $\sigma_{\max}(A)$ denotes the largest eigenvalue of the matrix A.

To show that the statement does not hold in general, it suffices to provide a counterexample. First, we consider the case where $\|\nabla \mathcal{L}_{\mathcal{D}_i}(\theta)\| = 0$ for all $i = 1, 2, \ldots, S$. Then, the zeroth-order sharpness of the i-th individual loss function is given by

$$\mathcal{S}_i(\theta) = \frac{1}{2}\rho^2 \sigma_{\max}(H_i(\theta)) + \mathcal{O}(\|\rho\|^3).$$

This leads to the following expression of the average sharpness over all individual loss functions:

$$\frac{1}{S}\sum_{i=1}^{S} \mathcal{S}_i(\theta) = \frac{1}{2S}\rho^2 \sum_{i=1}^{S} \sigma_{\max}(H_i(\theta)) + \mathcal{O}(\|\rho\|^3).$$

Next, consider two different local minima θ_1 and θ_2. For sufficiently small ρ, we can write:

$$\mathcal{S}_{\mathcal{D}_s}(\theta_1) < \mathcal{S}_{\mathcal{D}_s}(\theta_2) \tag{A.1}$$

$$\Leftrightarrow$$

$$\sigma_{\max}\left(\sum_{i=1}^{S} H_i(\theta_1)\right) < \sigma_{\max}\left(\sum_{i=1}^{S} H_i(\theta_2)\right). \tag{A.2}$$

Similarly, for sufficiently small ρ, we have the following relationship between the average sharpnesses at θ_1 and θ_2:

$$\frac{1}{S}\sum_{i=1}^{S} \mathcal{S}_i(\theta_1) < \frac{1}{S}\sum_{i=1}^{S} \mathcal{S}_i(\theta_2) \tag{A.3}$$

$$\Leftrightarrow$$

$$\sum_{i=1}^{S} \sigma_{\max}(H_i(\theta_1)) < \sum_{i=1}^{S} \sigma_{\max}(H_i(\theta_2)). \tag{A.4}$$

Consequently, we conclude that Eq. (A.1) does not imply Eq. (A.3) since the largest eigenvalue of a sum of matrices, $\sigma_{\max}\left(\sum_{i=1}^{S} H_i(\theta)\right)$, is not generally equal to the sum of the largest eigenvalues of the individual matrices, $\sum_{i=1}^{S} \sigma_{\max}(H_i(\theta))$.

Secondly, let us consider the case where $\nabla \mathcal{L}_{\mathcal{D}_s}(\theta) = 0$, but there exists at least two elements such that $\nabla \mathcal{L}_{\mathcal{D}_i}(\theta) \neq 0$. For simplicity, let $S = 2$. Without loss of generality, assume $\nabla \mathcal{L}_{\mathcal{D}_1}(\theta) > 0$ and $\nabla \mathcal{L}_{\mathcal{D}_2}(\theta) = -\nabla \mathcal{L}_{\mathcal{D}_1}(\theta)$. Then, the sharpness for $\mathcal{L}_{\mathcal{D}_1}(\theta)$ is given by

$$\mathcal{S}_{\mathcal{D}_1}(\theta) = \|\nabla \mathcal{L}_{\mathcal{D}_1}(\theta)\|\rho + \mathcal{O}(\|\rho\|^2).$$

Now, consider two local minima θ_1 and θ_2 satisfying the following inequality:

$$\mathcal{S}_{\mathcal{D}_s}(\theta_1) < \mathcal{S}_{\mathcal{D}_s}(\theta_2).$$

A counterexample can be constructed such that for some $G > 0$ and $0 < c < 1$,

$$\nabla \mathcal{L}_{\mathcal{D}_1}(\theta_1) = G = -\nabla \mathcal{L}_{\mathcal{D}_2}(\theta_1),$$

and

$$\nabla \mathcal{L}_{\mathcal{D}_1}(\theta_2) = cG = -\nabla \mathcal{L}_{\mathcal{D}_2}(\theta_2).$$

In this example, we find that $\frac{1}{S}\sum_{i=1}^{S} \mathcal{S}_i(\theta_1) > \frac{1}{S}\sum_{i=1}^{S} \mathcal{S}_i(\theta_2)$. However, such a choice of gradients does not affect the Hessian matrices, and thus the inequality for the sharpness of the total loss remains unchanged. Therefore, the sharpness for the total loss does not generally follow the same ordering as the average sharpness of the individual losses.

□

Appendix B. Comparison of two term in Eq. 4.2

Figure 10 shows that the second term tends to be slightly smaller than the first term, but the two are comparable in magnitude. This indicates that both terms contribute to the gradual perturbation.

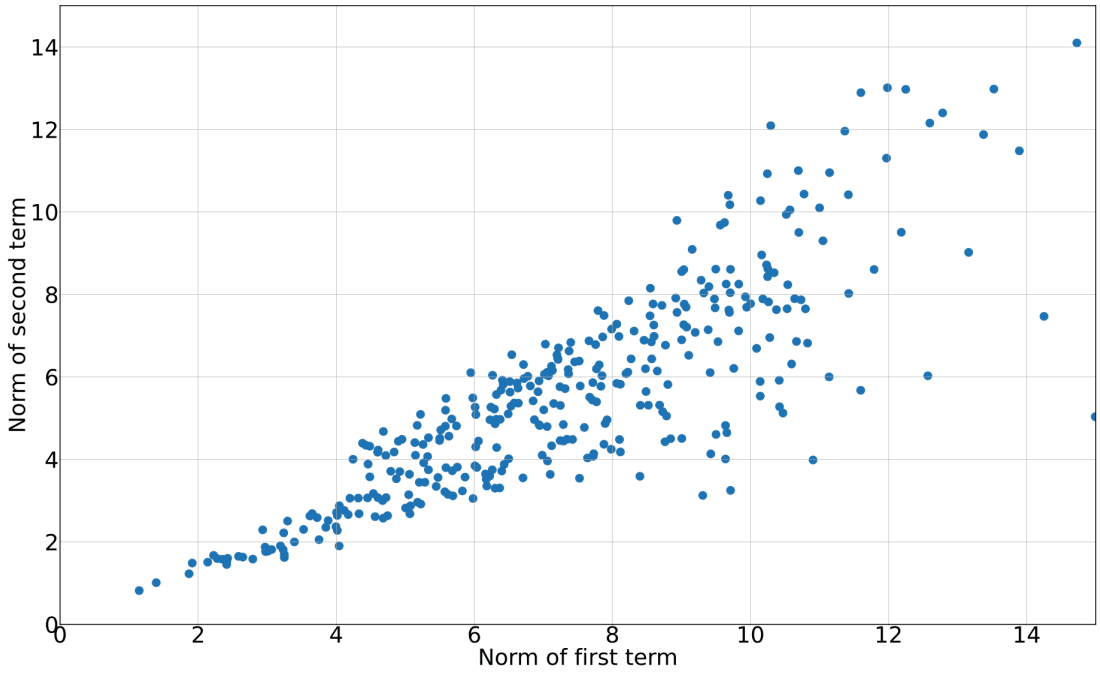
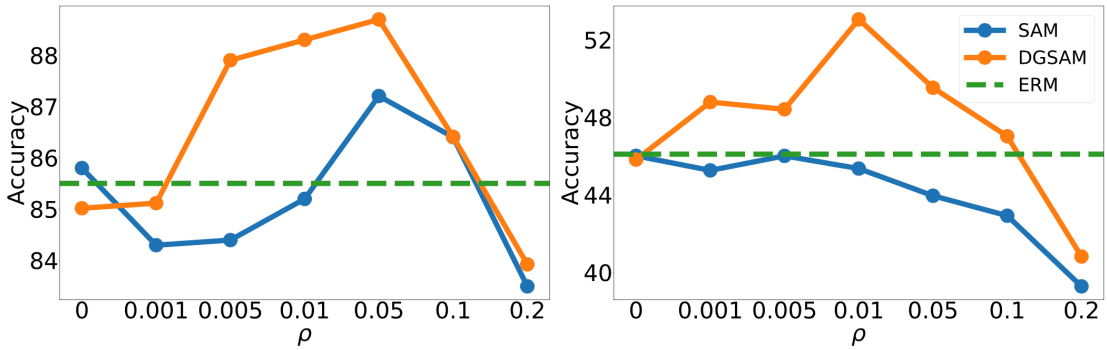


Figure 10. Comparison of magnitude of two term in Eq. 4.2 on the PACS

doi.org/10.32388/CLU2QF

Appendix C. Sensitivity Analysis

To analyze the sensitivity of DGSAM to ρ, we evaluated the performance of SAM and DGSAM across different ρ values $\{0.001, 0.005, 0.01, 0.05, 0.1, 0.2\}$ on the PACS and TERRAINCOGNITA datasets. As shown in Figure 11, DGSAM consistently outperformed SAM and demonstrated superior performance over a wider range of ρ values.



(a) PACS (b) TerraIncognita

Figure 11. Sensitivity analysis

Appendix D. Details of Experiments

D.1. Implementation Details

We searched hyperparameters in the following ranges: the learning rate was chosen from $\{10^{-5}, 2 \times 10^{-5}, 3 \times 10^{-5}, 5 \times 10^{-5}\}$, the dropout rate from $\{0.0, 0.2, 0.5\}$, the weight decay from $\{10^{-4}, 10^{-6}\}$, and ρ from $\{0.03, 0.05, 0.1\}$. Each experiment was repeated three times, using 20 randomly initialized models sampled from this space, following the DomainBed protocol[24]. The optimal hyperparameters selected based on DomainBed criteria for each dataset are provided in Table 3 to ensure replicability. All our experiments were conducted on an NVIDIA A100 GPU, using Python 3.11.5, PyTorch 2.0.0, Torchvision 0.15.1, and CUDA 11.7.

Dataset	Learning Rate	Dropout Rate	Weight Decay	ρ
PACS	3×10^{-5}	0.5	10^{-4}	0.03
VLCS	10^{-5}	0.5	10^{-4}	0.03
OfficeHome	10^{-5}	0.5	10^{-6}	0.1
TerraIncognita	10^{-5}	0.2	10^{-6}	0.05
DomainNet	2×10^{-5}	0.5	10^{-4}	0.1

Table 3. Optimal hyperparameter settings for each dataset

D.2. Full Results

Here are the detailed results of the main experiment in Section 5.2 for each dataset. The outcomes are marked with † if sourced from [19], ‡ if sourced from [56], and are unlabeled if sourced from individual papers. We note that all results were conducted in the same experimental settings as described in their respective papers. The value shown next to the performance for each test domain represents the standard error across three trials.

Algorithm	A	C	P	S	Avg	SD	(s/iter)
CDANN† [63]	84.6±1.8	75.5±0.9	96.8±0.3	73.5±0.6	82.6	9.2	0.11
IRM† [5]	84.8±1.3	76.4±1.1	96.7±0.6	76.1±1.0	83.5	8.4	0.12
DANN† [7]	86.4±0.8	77.4±0.8	97.3±0.4	73.5±2.3	83.7	9.2	0.11
MTL† [65]	87.5±0.8	77.1±0.5	96.4±0.8	77.3±1.8	84.6	8.0	0.12
VREx† [62]	86.0±1.6	79.1±0.6	96.9±0.5	77.7±1.7	84.9	7.6	0.11
MLDG† [42]	85.5±1.4	80.1±1.7	97.4±0.3	76.6±1.1	84.9	7.9	0.13
ARM† [43]	86.8±0.6	76.8±0.5	97.4±0.3	79.3±1.2	85.1	8.0	0.11
RSC† [64]	85.4±0.8	79.7±1.8	97.6±0.3	78.2±1.2	85.2	7.6	0.14
ERM†	84.7±0.4	80.8±0.6	97.2±0.3	79.3±1.0	85.5	7.0	0.11
CORAL† [67]	88.3±0.2	80.0±0.5	97.5±0.3	78.8±1.3	86.2	7.5	0.12
SagNet† [66]	87.4±1.0	80.7±0.6	97.1±0.1	80.0±0.4	86.3	6.9	0.32
SWAD [13]	89.3±0.2	83.4±0.6	97.3±0.3	82.5±0.5	88.1	5.9	0.11
SAM† [18]	85.6±2.1	80.9±1.2	97.0±0.4	79.6±1.6	85.8	6.9	0.22
GSAM† [50]	86.9±0.1	80.4±0.2	97.5±0.0	78.7±0.8	85.9	7.4	0.22
Lookbehind-SAM [53]	86.8±0.2	80.2±0.3	97.4±0.8	79.7±0.2	86.0	7.2	0.50
GAM‡ [51]	85.9±0.9	81.3±1.6	98.2±0.4	79.0±2.1	86.1	7.4	0.43
SAGM [19]	87.4±0.2	80.2±0.3	98.0±0.2	80.8±0.6	86.6	7.2	0.22
DISAM [21]	87.1±0.4	81.9±0.5	96.2±0.3	83.1±0.7	87.1	5.6	0.33
FAD [56]	88.5±0.5	83.0±0.8	98.4±0.2	82.8±0.9	88.2	6.3	0.38
DGSAM (Ours)	88.9±0.2	84.8±0.7	96.9±0.2	83.5±0.3	88.5	5.2	0.17
DGSAM + SWAD	89.1±0.5	84.6±0.4	97.3±0.1	83.6±0.4	88.7	5.4	0.17

Table 4. The performance of DGSAM with 20 baseline algorithms on PACS.

Algorithm	C	L	S	V	Avg	SD	(s/iter)
RSC†[64]	97.9±0.1	62.5±0.7	72.3±1.2	75.6±0.8	77.1	13.0	0.13
MLDG†[42]	97.4±0.2	65.2±0.7	71.0±1.4	75.3±1.0	77.2	12.2	0.12
MTL†[65]	97.8±0.4	64.3±0.3	71.5±0.7	75.3±1.7	77.2	12.5	0.12
ERM†	98.0±0.3	64.7±1.2	71.4±1.2	75.2±1.6	77.3	12.5	0.11
CDANN†[63]	97.1±0.3	65.1±1.2	70.7±0.8	77.1±1.5	77.5	12.1	0.11
ARM†[43]	98.7±0.2	63.6±0.7	71.3±1.2	76.7±0.6	77.6	13.1	0.11
SagNet†[66]	97.9±0.4	64.5±0.5	71.4±1.3	77.5±0.5	77.8	12.5	0.32
VREx†[62]	98.4±0.3	64.4±1.4	74.1±0.4	76.2±1.3	78.3	12.4	0.11
DANN†[7]	99.0±0.3	65.1±1.4	73.1±0.3	77.2±0.6	78.6	12.6	0.11
IRM†[5]	98.6±0.1	64.9±0.9	73.4±0.6	77.3±0.9	78.6	12.4	0.12
CORAL†[67]	98.3±0.1	66.1±1.2	73.4±0.3	77.5±1.2	78.8	12.0	0.12
SWAD[13]	98.8±0.1	63.3±0.3	75.3±0.5	79.2±0.6	79.1	12.8	0.11
GAM‡[51]	98.8±0.6	65.1±1.2	72.9±1.0	77.2±1.9	78.5	12.5	0.43
Lookbehind-SAM[53]	98.7±0.6	65.1±1.1	73.1±0.4	78.7±0.9	78.9	12.4	0.50
FAD[56]	99.1±0.5	66.8±0.9	73.6±1.0	76.1±1.3	78.9	12.1	0.38
GSAM†[50]	98.7±0.3	64.9±0.2	74.3±0.0	78.5±0.8	79.1	12.3	0.22
SAM†[18]	99.1±0.2	65.0±1.0	73.7±1.0	79.8±0.1	79.4	12.5	0.22
DISAM[21]	99.3±0.0	66.3±0.5	81.0±0.1	73.2±0.1	79.9	12.3	0.33
SAGM[19]	99.0±0.2	65.2±0.4	75.1±0.3	80.7±0.8	80.0	12.3	0.22
DGSAM + SWAD	99.3±0.7	67.2±0.3	77.7±0.6	79.2±0.5	80.9	11.6	0.17
DGSAM (Ours)	99.0±0.5	67.0±0.5	77.9±0.5	81.8±0.4	81.4	11.5	0.17

Table 5. The performance of DGSAM with 20 baseline algorithms on VLCS

Algorithm	A	C	P	R	Avg	SD	(s/iter)
IRM† [5]	58.9±2.3	52.2±1.6	72.1±2.9	74.0±2.5	64.3	9.1	0.12
ARM† [43]	58.9±0.8	51.0±0.5	74.1±0.1	75.2±0.3	64.8	10.2	0.11
RSC† [64]	60.7±1.4	51.4±0.3	74.8±1.1	75.1±1.3	65.5	10.0	0.14
CDANN† [63]	61.5±1.4	50.4±2.4	74.4±0.9	76.6±0.8	65.7	10.6	0.11
DANN† [7]	59.9±1.3	53.0±0.3	73.6±0.7	76.9±0.5	65.9	9.8	0.11
MTL† [65]	61.5±0.7	52.4±0.6	74.9±0.4	76.8±0.4	66.4	10.0	0.12
VREx† [62]	60.7±0.9	53.0±0.9	75.3±0.1	76.6±0.5	66.4	9.9	0.11
ERM†	61.3±0.7	52.4±0.3	75.8±0.1	76.6±0.3	66.5	10.2	0.11
MLDG† [42]	61.5±0.9	53.2±0.6	75.0±1.2	77.5±0.4	66.8	9.9	0.13
ERM†	63.1±0.3	51.9±0.4	77.2±0.5	78.1±0.2	67.6	10.8	0.11
SagNet† [66]	63.4±0.2	54.8±0.4	75.8±0.4	78.3±0.3	68.1	9.5	0.32
CORAL† [67]	65.3±0.4	54.4±0.5	76.5±0.1	78.4±0.5	68.7	9.6	0.12
SWAD [13]	66.1±0.4	57.7±0.4	78.4±0.1	80.2±0.2	70.6	9.2	0.11
GAM‡ [51]	63.0±1.2	49.8±0.5	77.6±0.6	82.4±1.0	68.2	12.8	0.43
FAD [56]	63.5±1.0	50.3±0.8	78.0±0.4	85.0±0.6	69.2	13.4	0.40
Lookbehind-SAM [53]	64.7±0.3	53.1±0.8	77.4±0.5	81.7±0.7	69.2	11.2	0.50
GSAM† [50]	64.9±0.1	55.2±0.2	77.8±0.0	79.2±0.0	69.3	9.9	0.22
SAM† [18]	64.5±0.3	56.5±0.2	77.4±0.1	79.8±0.4	69.6	9.5	0.22
SAGM [19]	65.4±0.4	57.0±0.3	78.0±0.3	80.0±0.2	70.1	9.4	0.22
DISAM [21]	65.8±0.2	55.6±0.2	79.2±0.2	80.6±0.1	70.3	10.3	0.33
DGSAM (Ours)	65.6±0.4	59.7±0.2	78.0±0.2	80.1±0.4	70.8	8.5	0.17
DGSAM + SWAD	66.2±0.6	59.9±0.1	78.1±0.4	81.2±0.5	71.4	8.7	0.17

Table 6. The performance of DGSAM with 20 baseline algorithms on OfficeHome

Algorithm	L100	L38	L43	L46	Avg	SD	(s/iter)
ARM† [43]	49.3±0.7	38.3±2.4	55.8±0.8	38.7±1.3	45.5	7.4	0.11
MTL† [65]	49.3±1.2	39.6±6.3	55.6±1.1	37.8±0.8	45.6	7.3	0.12
CDANN† [63]	47.0±1.9	41.3±4.8	54.9±1.7	39.8±2.3	45.8	5.9	0.11
ERM†	49.8±4.4	42.1±1.4	56.9±1.8	35.7±3.9	46.1	8.0	0.11
VREx† [62]	48.2±4.3	41.7±1.3	56.8±0.8	38.7±3.1	46.4	6.9	0.11
RSC† [64]	50.2±2.2	39.2±1.4	56.3±1.4	40.8±0.6	46.6	7.0	0.13
DANN† [7]	51.1±3.5	40.6±0.6	57.4±0.5	37.7±1.8	46.7	7.9	0.11
IRM† [5]	54.6±1.3	39.8±1.9	56.2±1.8	39.6±0.8	47.6	7.9	0.12
CORAL† [67]	51.6±2.4	42.2±1.0	57.0±1.0	39.8±2.9	47.7	7.0	0.12
MLDG† [42]	54.2±3.0	44.3±1.1	55.6±0.3	36.9±2.2	47.8	7.6	0.13
ERM†	54.3±0.4	42.5±0.7	55.6±0.3	38.8±2.5	47.8	7.3	0.11
SagNet† [66]	53.0±2.9	43.0±2.5	57.9±0.6	40.4±1.3	48.6	7.1	0.32
SWAD [13]	55.4±0.0	44.9±1.1	59.7±0.4	39.9±0.2	50.0	7.9	0.11
SAM† [18]	46.3±1.0	38.4±2.4	54.0±1.0	34.5±0.8	43.3	7.5	0.22
Lookbehind-SAM [53]	44.6±0.8	41.1±1.4	57.4±1.2	34.9±0.6	44.5	8.2	0.50
GAM‡ [51]	42.2±2.6	42.9±1.7	60.2±1.8	35.5±0.7	45.2	9.1	0.43
FAD [56]	44.3±2.2	43.5±1.7	60.9±2.0	34.1±0.5	45.7	9.6	0.38
DISAM [21]	46.2±2.9	41.6±0.1	58.0±0.5	40.5±2.2	46.6	6.9	0.33
GSAM† [50]	50.8±0.1	39.3±0.2	59.6±0.0	38.2±0.8	47.0	8.8	0.22
SAGM [19]	54.8±1.3	41.4±0.8	57.7±0.6	41.3±0.4	48.8	7.5	0.22
DGSAM (Ours)	53.8±0.6	45.0±0.7	59.1±0.4	41.8±1.0	49.9	6.9	0.17
DGSAM + SWAD	55.6±1.2	45.9±0.5	59.6±0.5	43.1±0.9	51.1	6.8	0.17

Table 7. The performance of DGSAM with 20 baseline algorithms on TerraIncognita

Algorithm	C	I	P	Q	R	S	Avg	SD	(s/iter)
VREx† [62]	47.3±3.5	16.0±1.5	35.8±4.6	10.9±0.3	49.6±4.9	42.0±3.0	33.6	15.0	0.18
IRM† [5]	48.5±2.8	15.0±1.5	38.3±4.3	10.9±0.5	48.2±5.2	42.3±3.1	33.9	15.2	0.19
ARM† [43]	49.7±0.3	16.3±0.5	40.9±1.1	9.4±0.1	53.4±0.4	43.5±0.4	35.5	16.7	0.18
CDANN†[63]	54.6±0.4	17.3±0.1	43.7±0.9	12.1±0.7	56.2±0.4	45.9±0.5	38.3	17.3	0.18
DANN† [7]	53.1±0.2	18.3±0.1	44.2±0.7	11.8±0.1	55.5±0.4	46.8±0.6	38.3	17.0	0.18
RSC† [64]	55.0±1.2	18.3±0.5	44.4±0.6	12.2±0.2	55.7±0.7	47.8±0.9	38.9	17.3	0.20
SagNet† [66]	57.7±0.3	19.0±0.2	45.3±0.3	12.7±0.5	58.1±0.5	48.8±0.2	40.3	17.9	0.53
MTL† [65]	57.9±0.5	18.5±0.4	46.0±0.1	12.5±0.1	59.5±0.3	49.2±0.1	40.6	18.4	0.20
ERM†	58.1±0.3	18.8±0.3	46.7±0.3	12.2±0.4	59.6±0.1	49.8±0.4	40.9	18.6	0.18
MLDG† [42]	59.1±0.2	19.1±0.3	45.8±0.7	13.4±0.3	59.6±0.2	50.2±0.4	41.2	18.4	0.34
CORAL†[67]	59.2±0.1	19.7±0.2	46.6±0.3	13.4±0.4	59.8±0.2	50.1±0.6	41.5	18.3	0.20
ERM†	62.8±0.4	20.2±0.3	50.3±0.3	13.7±0.5	63.7±0.2	52.1±0.5	43.8	19.7	0.18
SWAD† [13]	66.0±0.1	22.4±0.3	53.5±0.1	16.1±0.2	65.8±0.4	55.5±0.3	46.5	19.9	0.18
GAM‡ [51]	63.0±0.5	20.2±0.2	50.3±0.1	13.2±0.3	64.5±0.2	51.6±0.5	43.8	20.0	0.71
Lookbehind-SAM [53]	64.3±0.3	20.8±0.1	50.4±0.1	15.0±0.4	63.1±0.3	51.4±0.3	44.1	19.4	0.71
SAM† [18]	64.5±0.3	20.7±0.2	50.2±0.1	15.1±0.3	62.6±0.2	52.7±0.3	44.3	19.4	0.34
FAD [56]	64.1±0.3	21.9±0.2	50.6±0.3	14.2±0.4	63.6±0.1	52.2±0.2	44.4	19.5	0.56
GSAM† [50]	64.2±0.3	20.8±0.2	50.9±0.0	14.4±0.8	63.5±0.2	53.9±0.2	44.6	19.8	0.36
SAGM [19]	64.9±0.2	21.1±0.3	51.5±0.2	14.8±0.2	64.1±0.2	53.6±0.2	45.0	19.8	0.34
DISAM [21]	65.9±0.2	20.7±0.2	51.7±0.3	16.6±0.3	62.8±0.5	54.8±0.4	45.4	19.5	0.53
DGSAM (Ours)	63.6±0.4	22.2±0.1	51.9±0.3	15.8±0.2	64.7±0.3	54.7±0.4	45.5	19.4	0.26
DGSAM + SWAD	67.2±0.2	23.2±0.3	53.4±0.3	17.3±0.4	65.4±0.2	55.8±0.3	47.1	19.6	0.26

Table 8. The performance of DGSAM with 20 baseline algorithms on DomainNet

References

1. ^Kawaguchi K, Kaelbling LP, Bengio Y (2017). "Generalization in Deep Learning". arXiv preprint arXiv:1710.05468. arXiv:1710.05468.
2. ^Li H, Wang Y, Wan R, Wang S, Li TQ, Kot A (2020). "Domain Generalization for Medical Imaging Classification with Linear-Dependency Regularization". Advances in Neural Information Processing Systems. **33**: 3118–3129.
3. ^Khosravian A, Amirkhani A, Kashiani H, Masih-Tehrani M (2021). "Generalizing State-of-the-Art Object Detectors for Autonomous Vehicles in Unseen Environments". Expert Systems with Applications. **183**: 115417.
4. a, bMuandet K, Balduzzi D, Schölkopf B (2013). "Domain Generalization via Invariant Feature Representation". In: International conference on machine learning. pp. 10–18.
5. a, b, c, d, e, f, gArjovsky M, Bottou L, Gulrajani I, Lopez-Paz D (2019). "Invariant Risk Minimization". arXiv preprint arXiv:1907.02893. Available from: https://arxiv.org/abs/1907.02893.
6. a, bLi Y, Tian X, Gong M, Liu Y, Liu T, Zhang K, Tao D. Deep Domain Generalization via Conditional Invariant Adversarial Networks. In: Proceedings of the European conference on computer vision (ECCV). 2018. p. 624–639.
7. a, b, c, d, e, f, gGanin Y, Ustinova E, Ajakan H, Germain P, Larochelle H, Laviolette F, March M, Lempitsky V (2016). "Domain-Adversarial Training of Neural Networks". Journal of Machine Learning Research. **17** (59): 1–35.
8. ^Volpi R, Namkoong H, Sinha A, Duchi JC, Murino V. Generalizing to Unseen Domains via Adversarial Data Augmentation. In: Advances in Neural Information Processing Systems. 2018. p. 5334–5344.
9. ^Zhou K, Yang Y, Hospedales T, Xiang T. Learning to generate novel domains for domain generalization. In: Computer Vision--ECCV 2020: 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part XVI 16. Springer; 2020. p. 561-578.
10. ^Zhou K, Yang Y, Qiao Y, Xiang T (2021). "Domain Generalization with MixStyle". In: International Conference on Learning Representations (ICLR), 2021.
11. a, bLi Y, Yang Y, Zhou W, Hospedales T. Feature-Critic Networks for Heterogeneous Domain Generalization. In: International Conference on Machine Learning. PMLR; 2019. p. 3915-3924.
12. ^Balaji Y, Sankaranarayanan S, Chellappa R (2018). "MetaReg: Towards Domain Generalization using Meta-Regularization". In: Advances in Neural Information Processing Systems. pp. 998–1008.

13. a, b, c, d, e, f, g, h, i, j, k *Cha J, Chun S, Lee K, Cho HC, Park S, Lee Y, Park S (2021). "SWAD: Domain Generalizatio n by Seeking Flat Minima". In: Proceedings of the 35th International Conference on Neural Information Pro cessing Systems, pp. 22405–22418.*

14. a, b *Zhang Z, Luo R, Su Q, Sun X (2022). "GA-SAM: Gradient-Strength based Adaptive Sharpness-Aware Mini mization for Improved Generalization". Proceedings of the 2022 Conference on Empirical Methods in Natur al Language Processing. pp. 3888–3903.*

15. a, b *Jiang Z, Han X, Jin H, Wang G, Chen R, Zou N, Hu X (2023). "Chasing Fairness Under Distribution Shift: A Model Weight Perturbation Approach". Proceedings of the 37th International Conference on Neural Informa tion Processing Systems. pages 63931–63944.*

16. ^ *Peng D, Pan SJ (2022). "Learning gradient-based mixup towards flatter minima for domain generalizatio n". arXiv preprint arXiv:2209.14742. Available from: https://arxiv.org/abs/2209.14742.*

17. a, b *Chaudhari P, Choromanska A, Soatto S, LeCun Y, Baldassi C, Borgs C, Chayes J, Sagun L, Zecchina R. "Ent ropy-SGD: Biasing Gradient Descent into Wide Valleys". Journal of Statistical Mechanics: Theory and Experi ment. **2019** (12): 124018, 2019.*

18. a, b, c, d, e, f, g, h, i, j *Foret P, Kleiner A, Mobahi H, Neyshabur B. Sharpness-Aware Minimization for Efficiently Improving Generalization. In: International Conference on Learning Representations (ICLR); 2021.*

19. a, b, c, d, e, f, g, h, i, j, k, l, m, n, o, p *Wang P, Zhang Z, Lei Z, Zhang L. "Sharpness-Aware Gradient Matching for D omain Generalization." In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recog nition. 2023. p. 3769–3778.*

20. a, b, c *Shin S, Bae H, Na B, Kim YY, Moon I (2024). "Unknown Domain Inconsistency Minimization for Domai n Generalization." In: International Conference on Learning Representations (ICLR).*

21. a, b, c, d, e, f, g, h, i *Zhang R, Fan Z, Yao J, Zhang Y, Wang Y (2024). "Domain-Inspired Sharpness-Aware Minimi zation Under Domain Shifts". In: International Conference on Learning Representations (ICLR).*

22. ^ *Keskar NS, Mudigere D, Nocedal J, Smelyanskiy M, Tang PTP (2016). "On large-batch training for deep lear ning: Generalization gap and sharp minima". arXiv preprint arXiv:1609.04836. arXiv:1609.04836.*

23. a, b *Ghorbani B, Krishnan S, Xiao Y (2019). "An investigation into neural net optimization via hessian eigenv alue density". In: International Conference on Machine Learning. PMLR. pp. 2232–2241.*

24. a, b, c, d, e, f *Gulrajani I, Lopez-Paz D (2021). "In Search of Lost Domain Generalization". In: International Con ference on Learning Representations (ICLR).*

25. a, b *Li D, Yang Y, Song YZ, Hospedales TM (2017). "Deeper, Broader and Artier Domain Generalization". In: Pr oceedings of the IEEE international conference on computer vision. pp. 5542–5550.*

26. a, b Fang H, Siddiquie B, Siddiqui Y, Roy-Chowdhury AK, Davis LS. "Unbiased Metric Learning: On the Utilization of Multiple Datasets and Web Images for Softening Bias." In: Proceedings of the IEEE International Conference on Computer Vision. 2013. p. 1657–1664.

27. a, b, c Peng X, Bai Z, Xia X, Huang Z, Saenko K (2019). "Moment Matching for Multi-source Domain Adaptation". Proceedings of the IEEE/CVF International Conference on Computer Vision. 1406–1415.

28. ^ Koh PW, Sagawa S, Marklund H, Xie SM, Zhang M, Balsubramani A, Hu W, Yasunaga M, Phillips RL, Gao I, et al. WILDS: A Benchmark of in-the-Wild Distribution Shifts. In: International conference on machine learning. 2021. p. 5637-5664.

29. ^ Jia Y, Zhang J, Shan S, Chen X (2020). "Single-Side Domain Generalization for Face Anti-Spoofing". Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2020: 8484–8493.

30. ^ Akuzawa K, Iwasawa Y, Matsuo Y. Adversarial Invariant Feature Learning with Accuracy Constraint for Domain Generalization. In: Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2019, Würzburg, Germany, September 16–20, 2019, Proceedings, Part II. Springer; 2020. p. 315–331.

31. ^ Shao R, Lan X, Li J, Yuen PC. Multi-Adversarial Discriminative Deep Domain Generalization for Face Presentation Attack Detection. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2019. p. 10023–10031.

32. ^ Zhao S, Gong M, Liu T, Fu H, Tao D (2020). "Domain Generalization via Entropy Regularization". Advances in Neural Information Processing Systems. **33**: 16096–16107.

33. ^ Ghifary M, Balduzzi D, Kleijn WB, Zhang M (2016). "Scatter Component Analysis: A Unified Framework for Domain Adaptation and Domain Generalization". IEEE Transactions on Pattern Analysis and Machine Intelligence. **39** (7): 1414–1430.

34. ^ Li Y, Gong M, Tian X, Liu T, Tao D (2018). "Domain Generalization via Conditional Invariant Representations". In: Proceedings of the AAAI conference on artificial intelligence. **32** (1).

35. ^ Yoon C, Hamarneh G, Garbi R. Generalizable Feature Learning in the Presence of Data Bias and Domain Class Imbalance with Application to Skin Lesion Classification. In: Medical Image Computing and Computer Assisted Intervention--MICCAI 2019: 22nd International Conference, Shenzhen, China, October 13--17, 2019, Proceedings, Part IV 22. Springer; 2019. p. 365--373.

36. ^ Motiian S, Piccirilli M, Adjeroh DA, Doretto G (2017). "Unified Deep Supervised Domain Adaptation and Generalization". In: Proceedings of the IEEE international conference on computer vision. pp. 5715–5725.

37. ^Xu Z, Liu D, Yang J, Raffel C, Niethammer M. Robust and Generalizable Visual Representation Learning via Random Convolutions. In: International Conference on Learning Representations (ICLR); 2020.

38. ^Shi Y, Yu X, Sohn K, Chandraker M, Jain AK. "Towards Universal Representation Learning for Deep Face Recognition." In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2020. p. 6817–6826.

39. ^Qiao F, Zhao L, Peng X (2020). "Learning to Learn Single Domain Generalization". In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12556–12565.

40. ^Li D, Yang Y, Song YZ, Hospedales TM (2017). "Deeper, Broader and Artier Domain Generalization". In: Proceedings of the IEEE International Conference on Computer Vision. pp. 5542–5550.

41. ^Khosla A, Zhou T, Malisiewicz T, Efros AA, Torralba A. "Undoing the Damage of Dataset Bias." In: Computer Vision--ECCV 2012: 12th European Conference on Computer Vision, Florence, Italy, October 7-13, 2012, Proceedings, Part I 12. Springer; 2012. p. 158-171.

42. a, b, c, d, e, f, gLi D, Yang Y, Song YZ, Hospedales T (2018). "Learning to Generalize: Meta-Learning for Domain Generalization". Proceedings of the AAAI conference on artificial intelligence. **32** (1).

43. a, b, c, d, e, f, gZhang M, Marklund H, Dhawan N, Gupta A, Levine S, Finn C (2021). "Adaptive Risk Minimization: Learning to Adapt to Domain Shift". Advances in Neural Information Processing Systems. **34**: 23664–23678.

44. ^Seo S, Suh Y, Kim D, Kim G, Han J, Han B. Learning to Optimize Domain Specific Normalization for Domain Generalization. In: Computer Vision--ECCV 2020: 16th European Conference, Glasgow, UK, August 23--28, 2020, Proceedings, Part XXII 16. Springer; 2020. p. 68--83.

45. ^Xu Z, Li W, Niu L, Xu D. Exploiting Low-Rank Structure from Latent Domains for Domain Generalization. In: Computer Vision--ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part III 13. Springer; 2014. p. 628-643.

46. ^Hochreiter S, Schmidhuber J. (1994). "Simplifying Neural Nets by Discovering Flat Minima". In: Proceedings of the 7th International Conference on Neural Information Processing Systems. pp. 529–536.

47. ^Neyshabur B, Bhojanapalli S, McAllester D, Srebro N (2017). "Exploring Generalization in Deep Learning". In: Proceedings of the 31st International Conference on Neural Information Processing Systems. pp. 5949–5958.

48. ^Keskar NS, Mudigere D, Nocedal J, Smelyanskiy M, Tang PTP. "On Large-Batch Training for Deep Learning: Generalization Gap and Sharp Minima." In: International Conference on Learning Representations (ICLR); 2017.

49. ^Kwon J, Kim J, Park H, Choi IK. "ASAM: Adaptive Sharpness-Aware Minimization for Scale-Invariant Learning of Deep Neural Networks." In: International Conference on Machine Learning. PMLR; 2021. p. 5905-5914.

50. a, b, c, d, e, f, g, h, iZhuang J, Gong B, Yuan L, Cui Y, Adam H, Dvornek NC, Duncan JS, Liu T, et al. Surrogate Gap Minimization Improves Sharpness-Aware Training. In: International Conference on Learning Representations (ICLR); 2022.

51. a, b, c, d, e, f, gZhang X, Xu R, Yu H, Zou H, Cui P (2023). "Gradient Norm Aware Minimization Seeks First-Order Flatness and Improves Generalization". Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pages 20247–20257.

52. ^Zhang M, Lucas J, Ba J, Hinton GE (2019). "Lookahead Optimizer: K Steps Forward, 1 Step Back". Advances in Neural Information Processing Systems. **32**.

53. a, b, c, d, e, f, gMordido G, Malviya P, Baratin A, Chandar S. "Lookbehind-SAM: K Steps Back, 1 Step Forward." In: Forty-first International Conference on Machine Learning; 2024.

54. ^Du J, Yan H, Feng J, Zhou JT, Zhen L, Goh RSM, Tan V (2022). "Efficient Sharpness-Aware Minimization for Improved Training of Neural Networks". International Conference on Learning Representations (ICLR).

55. ^Liu Y, Mai S, Chen X, Hsieh CJ, You Y (2022). "Towards Efficient and Scalable Sharpness-Aware Minimization". Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pages 12360–12370.

56. a, b, c, d, e, f, g, h, i, jZhang X, Xu R, Yu H, Dong Y, Tian P, Cui P (2023). "Flatness-Aware Minimization for Domain Generalization". Proceedings of the IEEE/CVF International Conference on Computer Vision. 2023: 5189–5202.

57. a, bLe BM, Woo SS (2024). "Gradient Alignment for Cross-Domain Face Anti-Spoofing". Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pages 188–199.

58. a, bHe K, Zhang X, Ren S, Sun J (2016). "Deep residual learning for image recognition." In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778.

59. ^Ma H, Zhang Y, Sun S, Liu T, Shan Y (2023). "A comprehensive survey on NSGA-II for multi-objective optimization and applications". Artificial Intelligence Review. **56** (12): 15217–15270.

60. ^Luo H, Truong T, Pham T, Harandi M, Phung D, Le T (2024). "Explicit Eigenvalue Regularization Improves Sharpness-Aware Minimization". Advances in Neural Information Processing Systems. **37**: 4424–4453.

61. ^Wen K, Ma T, Li Z (2023). "How sharpness-aware minimization minimizes sharpness?" In: The eleventh international conference on learning representations.

62. a, b, c, d, e, f *Krueger D, Caballero E, Jacobsen JH, Zhang A, Binas J, Zhang D, Le Priol R, Courville A. Out-of-Distribution Generalization via Risk Extrapolation (Rex). In: International conference on machine learning. PMLR; 2021. p. 5815-5826.*

63. a, b, c, d, e, f *Li Y, Gong M, Tian X, Liu T, Tao D (2018). "Domain Generalization via Conditional Invariant Representations". In: Proceedings of the AAAI conference on artificial intelligence. 32(1).*

64. a, b, c, d, e, f *Huang Z, Wang H, Xing EP, Huang D. "Self-Challenging Improves Cross-Domain Generalization." In: Computer Vision--ECCV 2020: 16th European Conference, Glasgow, UK, August 23--28, 2020, Proceedings, Part II 16. Springer; 2020. p. 124--140.*

65. a, b, c, d, e, f *Blanchard G, Deshmukh AA, Dogan U, Lee G, Scott C (2021). "Domain Generalization by Marginal Transfer Learning". Journal of Machine Learning Research. 22 (2): 1–55.*

66. a, b, c, d, e, f *Nam H, Lee H, Park J, Yoon W, Yoo D (2021). "Reducing Domain Gap by Reducing Style Bias". Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2021: 8690–8699.*

67. a, b, c, d, e, f *Sun B, Saenko K. Deep CORAL: Correlation Alignment for Deep Domain Adaptation. In: Computer Vision--ECCV 2016 Workshops: Amsterdam, The Netherlands, October 8-10 and 15-16, 2016, Proceedings, Part III 14. Springer; 2016. p. 443-450.*

68. ^ *Venkateswara H, Eusebio J, Chakraborty S, Panchanathan S (2017). "Deep Hashing Network for Unsupervised Domain Adaptation". In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 5018–5027.*

69. ^ *Beery S, Van Horn G, Perona P (2018). "Recognition in Terra Incognita." In: Proceedings of the European conference on computer vision (ECCV), pp. 456–473.*

70. ^ *Kingma D, Ba J (2015). "Adam: {A} method for stochastic optimization". In: International Conference on Learning Representations (ICLR).*

Declarations

Funding: No specific funding was received for this work.

Potential competing interests: No potential competing interests to declare.