ECAI 2024
U. Endriss et al. (Eds.)

doi:10.3233/FAIA240782

Domain Feature Perturbation for Domain Generalization

Chenguang Wang[a], Zijun Zhang[a,*] and Zhidan Zhou[a]

[a]School of Cyber Science and Engineering, Wuhan University

Abstract. Deep neural networks (DNNs) often struggle with distribution shifts between training and test environments, which can lead to poor performance, untrustworthy predictions, or unexpected behaviors. This work proposes Domain Feature Perturbation (DFP), a novel approach that explicitly leverages domain information to improve the out-of-distribution performance of DNNs. Specifically, DFP trains a domain classifier in conjunction with the main prediction model and perturbs the multi-layer representation of the latter with random noise modulated by the gradient of the former. The domain classifier is designed to share the backbone with the main model and is easy to implement with minimal extra model parameters that can be discarded at inference time. Intuitively, the proposed method aims to reduce the dependence of the main prediction model on domain-specific features, such that the model can focus on domain-agnostic features that generalize across different domains. The results demonstrate the effectiveness of DFP on multiple benchmarks for domain generalization. Our code is available [39].

1 Introduction

Deep neural networks (DNNs) have exhibited impressive performance in solving a wide variety of real-world tasks. A crucial aspect of their success lies in their ability to learn from a large amount of training data [32]. However, despite the widely held assumption that training and test data are sampled from the same underlying probability distribution [10], real-world data frequently deviates from the training data distribution. Consequently, a shift occurs between the training and test data distributions. Such shifts are prevalent in diverse tasks; examples include face recognition under varying lighting conditions or backgrounds [1], medical image recognition with different imaging devices or acquisition protocols [52], and autonomous driving in different cities [34]. In these cases, the presence of out-of-distribution (OOD) data poses significant challenges to conventional supervised learning methods, leading to inaccurate and unreliable predictions.

To address the challenges of OOD generalization, researchers have explored various techniques, such as meta-learning [5], causal learning [24, 22], contrastive learning [15], and disentangled representation learning [49]. In particular, when given access to training data that is split into multiple domains and expected to generalize to an unseen test domain (without any information about them during training), which is known as *Domain Generalization*, one can leverage domain information (i.e., domain labels) to achieve better OOD performance. Domain Generalization and Domain Adaptation are both techniques used in machine learning to address the challenge of domain shift. Different from Domain Generalization, Domain Adaptation focuses on adapting a model to perform well on a specific, known target domain that is different from the source domains but is available during training. However, existing approaches to Domain Generalization have shown limited success in utilizing such information [11, 46] or relying on complex training procedures that involve adversarial training [9, 16, 21] or separate auxiliary models [21, 5, 49].

In this work, we propose a novel approach to leveraging domain information based on gradients. Specifically, we train a domain classifier in conjunction with the main prediction model and perturb the multi-layer representation of the latter with noise modulated by the gradient of the former. The domain classifier is designed to share the backbone with the main model, introducing minimal extra model parameters that can be discarded at inference time. This technique aims to reduce the dependence of the main prediction model on domain-specific features, allowing the model to focus on domain-agnostic features that generalize across different domains. The resulting method, named *Domain Feature Perturbation (DFP)*, is easy to implement without major modifications to the model architecture and shows significantly improved performance on multiple domain generalization benchmarks containing typical image data and graph data. We summarize our main contributions as follows:

- We explicitly leverage domain information by training a domain classifier along with the main model. We then compute the gradient of the domain classifier with respect to intermediate representations, the magnitude of which is utilized to identify domain-specific features in a relatively simple manner.
- We propose to perturb intermediate representations with random noise modulated by the aforementioned gradient magnitude. By doing so, domain-specific features are automatically identified and randomized, effectively reducing the dependence of the main model on these features. In addition, we propose a gradient similarity measure to evaluate the regularization effect.
- We evaluate our method on multiple image-based domain generalization benchmarks following a recently proposed evaluation protocol [46] for a fair comparison with other methods. In addition to the image datasets, we also validate our method on the graph data. The experimental results demonstrate competitive or better performance compared to state-of-the-art methods.

The rest of the paper is organized as follows. Section 2 discusses the related work. Section 3 develops the proposed method. Section 4 evaluates the method on various datasets, and Section 5 concludes the paper.

* Corresponding Author. Email: zijunzhang@whu.edu.cn

2 Related work

In this section, we review existing approaches to domain generalization and discuss their connections to our work.

2.1 Adversarial Training

Adversarial training serves as a key technique in enhancing the robustness of DNNs to distributional shifts in data. It is particularly employed to improve OOD generalization by encouraging the learning of domain-invariant features [25, 16]. In addition, applying adversarial perturbations to intermediate representations has been explored as a regularization technique [29, 48, 47, 41]. However, adversarial training can often lead to stability issues or slow convergence during training, and learning domain-invariant features may suppress useful features for other tasks in multi-task learning settings. In contrast, our proposed domain feature perturbation discourages the dependence of the main classifier on domain-specific features, while keeping the ability of the model to extract those features. Furthermore, it generates the perturbations without adversarial training, resulting in a simpler and more stable training process.

The concept of generative adversarial training extends to the use of domain classifiers in addressing domain generalization challenges. Adopting the idea of joint disentanglement and generative adversarial training, Wu et al. [43] attempt to separate irrelevant information by constraining the domain discriminator. Zhu et al. [53] incorporate a localized domain classifier with adversarial learning to produce locally mixed domains. Our method also integrates a domain classifier; however, it diverges from generative adversarial constraints and does not focus on identifying unseen domain data.

2.2 Conventional Methods

Data augmentation, extraction of invariant representations, and general regularization techniques collectively contribute to a broader understanding of domain generalization, offering diverse perspectives and solutions to the challenges in this field.

Data augmentation is a prevalent technique to enhance data diversity in various machine learning tasks [33, 45]. In the context of domain generalization, existing methods often emphasize feature-level augmentation [18]. Mixstyle [51] generates new styles by blending feature statistics from two instances with random convex weights. Xu et al. [44] employ Fourier-based data augmentation, consolidating information from multiple source domains. Other approaches, including domain randomization [35, 13, 8], introduce perturbations to data statistics to simulate diverse visual styles. We note that DFP is loosely related to domain randomization in that it also aims to randomize domain-specific features. Nevertheless, the perturbations introduced by DFP are applied to intermediate representations and are automatically generated with a domain classifier.

Another challenge for OOD generalization is the tendency of DNNs to overly specialize in extracting features that are useful for the training data. As such, several strategies have been developed to cultivate robust representations. These include invariant representations [26], meta-learning [23, 5, 50], transfer learning [4], representation disentanglement [49], model calibration [38], and causal learning [24, 22]. Contrary to learning invariant representation, Shi et al. [31] and Rame et al. [27] focus on gradient invariance during training. While DFP does not explicitly learn invariant representations, its carefully crafted perturbation encourages more dependence on invariant representations as discussed in **Section 3.2**. This distinction makes DFP potentially better suited for multitask learning, which requires the extraction of diverse features.

In addition to data augmentation and invariant learning, more general regularization techniques have been explored to tackle the OOD challenge, including ensemble learning [2, 30, 19], sharpness-aware optimization [6, 54], and several others [28, 15, 7].

3 Methods

In this section, we first explain the basic concepts crucial to our approach, including out-of-distribution (OOD) generalization and diversity shift. Subsequently, we present our method in detail and define gradient similarity for the purpose of evaluation. Figure 1 provides an overview of the framework of our method.

3.1 Preliminaries

3.1.1 OOD generalization

We begin by framing the general OOD generalization problem. Let X denote the input space and Y the target label space. We define a parametric model $f_\theta : X \to Y$, mapping input features to labels with parameters θ. The loss function $L : \theta \times (X \times Y) \to \mathbb{R}$ quantifies the discrepancy between the predicted and true labels. Given a supervised learning task with N training samples, $\{(x_i, y_i)\}_{i=1}^N$, where $x_i \in X$ and $y_i \in Y$, and sourced from the training distribution $P_{tr}(X, Y)$, the objective of OOD generalization is to identify a model that generalizes effectively to data from the test distribution $P_{te}(X, Y)$. However, without access to $P_{te}(X, Y)$ at training time, the model is usually optimized to minimize the empirical risk on $P_{tr}(X, Y)$:

$$\hat{f}_\theta = \arg\min_\theta \mathbb{E}_{X,Y\sim P_{tr}} L\left(f_\theta(X), Y\right). \tag{1}$$

In standard supervised learning, it's typically assumed that both training and test samples are i.i.d. samples from a shared distribution, denoted as $P_{tr}(X, Y) = P_{te}(X, Y)$. However, in the broader OOD context, training and test data may originate from different distributions, implying $P_{tr}(X, Y) \neq P_{te}(X, Y)$. For domain generalization tasks in particular, the combined training and test data are partitioned into k domains, represented as $D = \{D_i\}_{i=1}^k$, with each domain stemming from a unique distribution. During training, $k - 1$ of these domains form the training set, while the remaining domain is held out for testing.

3.1.2 Diversity shift

Distribution shifts between training and test data can be categorized into diversity shift and correlation shift [46]. Consider a training distribution $P_{tr}(X, Y)$ and a test distribution $P_{te}(X, Y)$ with probability functions p and q, respectively. The labeling rule of the data, $f : X \to Y$, usually depends on a particular set of features $\mathcal{Z}_1$, whereas the rest of the features $\mathcal{Z}_2$ are not causal to the prediction of Y. That is, while both $\mathcal{Z}_1$ and $\mathcal{Z}_2$ jointly determine the input variable X, the target variable Y is determined by $\mathcal{Z}_1$ alone. The following property for every $z \in \mathcal{Z}_1$ makes OOD generalization possible:

$$p(z) \cdot q(z) \neq 0 \wedge \forall y \in Y : p(y|z) = q(y|z). \tag{2}$$

And the opposite property of $z \in \mathcal{Z}_2$ makes OOD generalization challenging:

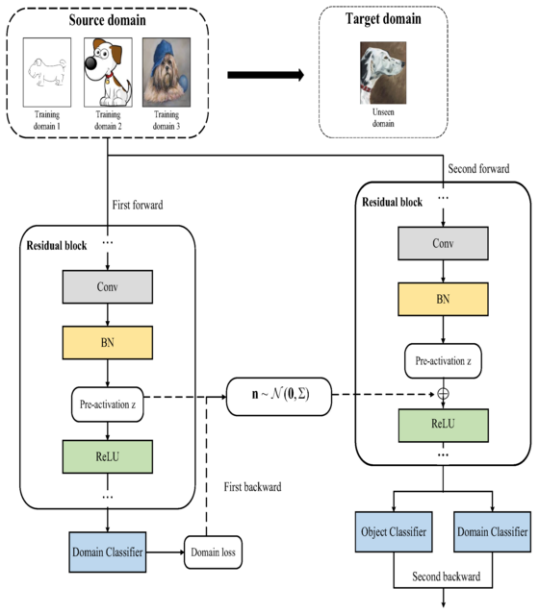


Figure 1. Overview of the training procedure of DFP. The two classification heads share a single backbone network. The first backward pass generates the modulated perturbation, followed by a second backward pass that updates the model parameters.

$$p(z) \cdot q(z) = 0 \vee \exists y \in Y : p(y|z) \neq q(y|z). \tag{3}$$

Diversity shift further assumes that $p(z) \cdot q(z) = 0$ for $z \in \mathcal{Z}_2$, meaning that the diversity of data is embodied by unique features not shared by the different domains. These features are defined as $\mathcal{S} = \{z \in \mathcal{Z}_2 | p(z) \cdot q(z) = 0\}$. The extent of diversity shift can be measured by

$$D_{div}(p, q) := \frac{1}{2} \int_{\mathcal{S}} |p(z) - q(z)| \, \mathrm{d}z. \tag{4}$$

It is observed that many practical datasets for domain generalization exhibit significant diversity shifts [46]. Therefore, exploiting the property of diversity shift may improve OOD generalization on such datasets.

3.2 Domain Feature Perturbation

Given sufficient capacity, a parametric model f_θ trained with the objective in **Equation 1** is expected to capture the causal features in $\mathcal{Z}_1$, which is desirable for OOD generalization. However, f_θ may also rely on the non-causal features in $\mathcal{Z}_2$ if they correlate with Y. While the dependency on non-causal features can aid in i.i.d. generalization in the training environment, it is unlikely to generalize to the unseen test environment, especially considering the non-overlapping supports of $\mathcal{Z}_2$ caused by diversity shift (i.e. $p(z) \cdot q(z) = 0$ for $z \in \mathcal{Z}_2$). Theoretically, one can reduce the dependence on $\mathcal{Z}_2$ to achieve better OOD generalization. However, there is no guarantee that features from $\mathcal{Z}_1$ and $\mathcal{Z}_2$ captured by f_θ are well-separated, and identifying them is even more challenging. To tackle this challenge, we observe that, compared to $\mathcal{Z}_1$, the non-overlapping supports of $\mathcal{Z}_2$ across different domains make it particularly useful for domain classification. As such, we propose to train a domain classifier to differentiate among different training domains, and use its gradient to approximately identify the features in $\mathcal{Z}_2$. Subsequently, we perturb the identified features with noise, reducing the dependence of f_θ on them in the training process. We refer to this approach as *Domain Feature Perturbation (DFP)*.

Concretely, alongside the main classifier, we train a domain classifier to capture domain-specific features. The two classifiers can be implemented with two output heads that share the same backbone, and thus introduces negligible extra parameters. Let $g_\theta : X \to Y_d$ denote the domain classifier, where Y_d is the set of training domain labels. The overall loss function is as follows:

$$\mathcal{L}(\theta) = \mathbb{E}_{X,Y,Y_d \sim P_{tr}} \left[\alpha L \left(f_\theta(X), Y \right) + (1 - \alpha) L_d \left(g_\theta(X), Y_d \right) \right], \tag{5}$$

where L and L_d correspond respectively to the main classifier and the domain classifier, and $\alpha \in (0, 1)$ is a hyperparameter to weight the two losses. To utilize the gradient of the domain classifier, the training process of DFP involves two forward passes and two backward passes in each iteration. In the first forward pass and backward pass, we compute L_d, as well as its gradient with respect to the intermediate representations of the backbone network, i.e.,

$$\nabla_{\mathbf{z}} L_d = \frac{\partial L_d}{\partial \mathbf{z}}, \tag{6}$$

where $\mathbf{z} \in \mathbb{R}^m$ represents the pre-activations of neurons. Let z_i be the i-th element of $\mathbf{z}$, and ϵ a positive constant. The magnitude of $\nabla_{\mathbf{z}} L_d$ then serves to modulate a random noise vector $\mathbf{n} \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma})$, such that

$$\mathbf{\Sigma} = \mathrm{diag}\left(\sigma_1^2, \sigma_2^2, \ldots, \sigma_m^2\right), \text{and } \sigma_i = \frac{\epsilon}{\|\nabla_z L_d\|_p} \left| \frac{\partial L_d}{\partial z_i} \right|. \tag{7}$$

In the second forward pass, DFP applies modulated noise $\mathbf{n}$ to $\mathbf{z}$ at each layer as $\tilde{\mathbf{z}} = \mathbf{z} + \mathbf{n}$, and proceeds with the second backward pass to update model parameters. Note that in **Equation 7**, the σ_i's are normalized by the ℓ_p norm of $\nabla_z L_d$ to keep the overall magnitude of $\mathbf{\Sigma}$ stable. In addition, p is set to 2 throughout this paper unless otherwise specified. At test time, the domain classifier can be discarded, and the inference of the main classifier can be done in a single forward pass.

Algorithm 1 Training procedure with domain feature perturbation (DFP)

Input: Main classifier f_θ; Domain classifier g_θ; Loss function $\mathcal{L}(\theta)$; Training data D_{tr}; Batch size B; Number of training epochs E.

Training:

for epoch $e = 1, 2, ..., E$ **do**
  for batch $b = b_1, b_2, ... \subset D_{tr}$ with batch size B **do**
    $b_i = \{x_i, y_i, y_{d,i}\}$
    $\hat{y}_{d,i} = g_\theta(x_i)$ ▷ First forward
    $\nabla_{\mathbf{z}} L_d = \partial L_d(\hat{y}_{d,i}, y_{d,i}) / \partial \mathbf{z}$ ▷ First backward
    $\mathbf{n} \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma})$ ▷ Sample DFP as per Equation 7
    $\hat{y}_i = f_\theta(x_i), \hat{y}_{d,i} = g_\theta(x_i)$ with $\tilde{\mathbf{z}} = \mathbf{z} + \mathbf{n}$ ▷ Second forward
    $\mathcal{L}(\theta) = \alpha L(\hat{y}_i, y_i) + (1 - \alpha) L_d(\hat{y}_{d,i}, y_{d,i})$
    $\nabla_\theta \mathcal{L}(\theta) = \partial \mathcal{L}(\theta) / \partial \theta$ ▷ Second backward
    $\theta \leftarrow \mathrm{Optimizer}\left(\theta, \nabla_\theta \mathcal{L}(\theta)\right)$
  end for
end for

The intuition behind Equation 7 is that the features in $\mathcal{Z}_2$ are predominantly utilized by the domain classifier compared to those in

$\mathcal{Z}_1$, and thus are more likely to have larger gradient magnitudes, $|\partial L_d/\partial z_i|$. Analogous to Fisher information, which is the variance of the score, or the expected value of the observed information, we maintain the noise mean at zero while modulating the noise variance in response to the domain gradient. Thereby emulating the correlation between the magnitude of noise and variance. By injecting noise with high variance into the features in $\mathcal{Z}_2$, we aim to diminish the reliance of f_θ on these features, thereby improving OOD generalization. The training procedure with DFP is detailed in **Algorithm 1**.

3.3 Gradient Similarity

As discussed in **Section 3.2**, a model trained with DFP is expected to focus more on the features in $\mathcal{Z}_1$ than those in $\mathcal{Z}_2$. To verify this, one can compare the magnitudes of L's gradients with respect to the two sets of features, which again requires separating and identifying the features in $\mathcal{Z}_1$ and $\mathcal{Z}_2$. To circumvent this requirement, we observe that, compared to $\mathcal{Z}_2$, the model relies on $\mathcal{Z}_1$ should have similar gradients across different domains, since $\mathcal{Z}_1$ is domain-agnostic while $\mathcal{Z}_2$ is domain-specific. Therefore, we propose a simple gradient similarity measure to evaluate how much a model relies on domain-agnostic features. Specifically, given k different domains, $D = \{D_i\}_{i=1}^k$, we denote the mean absolute gradient of L with respect to $\mathbf{z}$ for each domain as

$$\eta_i = \frac{1}{|D_i|} \sum\nolimits_{(x.y)\in D_i} \text{Abs}\left(\frac{\partial L\left(f_\theta(x), y\right)}{\partial \mathbf{z}}\right), \tag{8}$$

then the gradient similarity on D is defined as

$$\begin{aligned} \mathcal{S}(D) &= \frac{2\sum_{1\le i<j\le k} \text{CosSim}\left(\eta_i, \eta_j\right)}{k(k-1)}, \\ \text{where } \text{CosSim}\left(\boldsymbol{\mu}, \boldsymbol{\nu}\right) &= \frac{\boldsymbol{\mu}\cdot\boldsymbol{\nu}}{\|\boldsymbol{\mu}\|\|\boldsymbol{\nu}\|}. \end{aligned} \tag{9}$$

Intuitively, **Equation 9** calculates the cosine similarity between every possible pair of η_i and η_j that are derived from two different domains, and averages the similarity values together.

4 Experiments

4.1 Experimental Setup

In order to validate the effectiveness of our method, we mainly test it on two types of data: general image data and graph data.

4.1.1 Image datasets

We evaluate our proposed method on three domain generalization datasets with image data: PACS [17] with four artistic styles and seven categories, OfficeHome [37] with 4 artistic styles and 65 categories, and Terra Incognita [3] with 4 camera locations and 10 categories. As noted by Ye et al. [46], these datasets exhibit significant diversity shifts and thus are suitable for our evaluation. Figure 2 illustrates several images from the PACS dataset. Figure 3 also shows the cosine similarity values across different domains in the PACS dataset. It is readily apparent that there are significant differences in the data distribution across domains.

(a) Art-painting (b) Cartoon (c) Photo (d) Sketch

Figure 2. Illustration of images of the dog category in different domains in the PACS dataset.

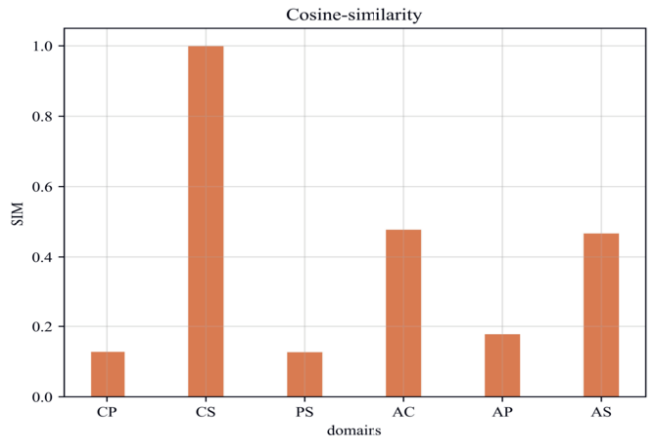


Figure 3. Cosine similarity between different domains on PACS.

4.1.2 Graph datasets

We investigate the scenario where there are multiple observed graphs in one dataset, and a model trained on a subset of them is expected to generalize to unseen graphs. The graphs in a dataset share the same input feature space and output space, but can have different sizes and data distributions since they are collected from different domains. In our experiments, we use the public social network dataset Facebook-100 [20].

4.1.3 Model Selection Methods

There are three model selection strategies commonly used for domain generalization tasks: training-domain validation, test-domain validation, and leave-one-domain-out validation. Test-domain validation involves model selection where the validation set follows the same distribution as the test domain. It is important to adapt the degree of model invariance according to the test domain. The training-domain validation technique divides each training domain into training and validation subsets and selects the model with the highest accuracy based on the union of the validation subsets. This technique implies that the distributions of the training and test cases are similar. Both methods are reasonable and arguable, so in our experiments, we adopt training-domain validation and test-domain validation following the evaluation protocol used by Ye et al. [46], Gulrajani and Lopez-Paz [11], and Wang et al. [40]. To conduct a fair comparison, we compare our results with some results from existing benchmark work [46, 11]. We also run several methods that

were recently proposed, such as Sharpness-Aware Gradient Matching (SAGM) [40] (for image data) and Explore-to-Extrapolate Risk Minimization (EERM) [42] (for graph data), to demonstrate the effectiveness of our method. All experiments are conducted using PyTorch on Tesla V100 GPUs.

4.2 Domain Generalization On Image Data

In this section, we present the test accuracy of various methods on unseen domain image data to evaluate their domain generalization performance. We employ ResNet-18 [12] as the backbone architecture for most experiments. In addition, to further substantiate the effectiveness of our method, we also conduct experiments on the PACS dataset using ResNet-50 as the backbone. We also train the model without dropout.

For the PACS dataset, we perform eight rounds of random searching procedures for weight initialization, dataset division, and batch size. For the OfficeHome and Terra Incognita datasets, we perform three rounds of random searches for the basic hyperparameters. For all datasets, we train the model for the same number of steps and conduct three independent training runs to obtain the average results. We also evaluate the sensitivity of the proposed method to different loss weights $(\alpha, 1-\alpha)$, and the results are shown in Figure 4. We observe that a higher weight for the main classifier works better in practice. As such, we primarily use loss weights $(\alpha, 1-\alpha) \in \{(0.9, 0.1), (0.99, 0.01)\}$ for following experiments.

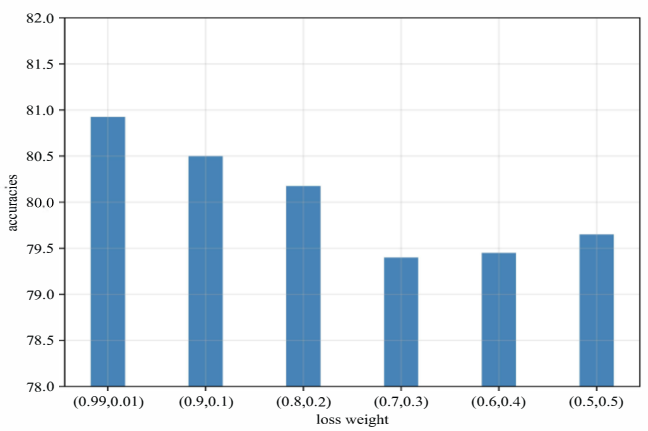


Figure 4. Accuracy results of different loss weights on PACS.

4.2.1 PACS

The PACS dataset comprises four training domain combinations {(C, P, S), (A, P, S), (A, C, S), (A, C, P)} that correspond to four test domains {A, C, P, S}. We set $\epsilon \in \{0.001, 0.005, 0.01, 0.05, 0.1, 0.2\}$ for different domain combinations in the PACS dataset. Table 1 presents the best results on the PACS dataset, with upward and downward arrows indicating increases and decreases in the data, respectively. Compared to Empirical Risk Minimization (ERM) [36], DFP improves the accuracy on every test domain, resulting in a 2.9% increase in the average accuracy. And DFP also surpasses the effect of the Representation Self-Challenging (RSC) [14] in overall average accuracy. It is worth noting that the improvement varies across different domains and is more significant in difficult domains such as A, C, and S.

Table 1. Test accuracies of ERM, RSC and DFP on PACS.

PACS	A	C	P	S	Avg
ERM	77.9 ± 0.9	72.7 ± 0.3	95.8 ± 0.2	74.1 ± 0.4	80.1
RSC	79.8 ± 0.2	**77.1 ± 0.7**	94.8 ± 0.1	76.3 ± 0.6	82.0 ↑ +1.9
DFP (Ours)	**80.6 ± 0.9**	75.8 ± 1.4	**96.2 ± 0.3**	**79.3 ± 1.0**	**83.0** ↑ +2.9

To compare the performance of different model sizes, we also repeat the experiments on the PACS dataset using Resnet-50 as the backbone. Following Cha et al. [6], we select the model with test-domain validation and train the model for the same number of steps. We compare DFP with state-of-the-art methods, including Fishr [27], SWAD [6], SAGM_DG [40], which also use Resnet-50 as the backbone. The best results are highlighted in Table 2 as part of out-domain test accuracy. Compared to ERM using the same backbone, DFP improves the average test accuracy by 1.1%, which is competitive to other state-of-the-art methods.

Table 2. Test accuracies on PACS with Resnet-50.

In/Out	Method	A	C	P	S	Avg
Out-domain	SWAD	86.1 ± 1.2	78.3 ± 3.6	97.3 ± 0.3	70 ± 3.3	82.9 ↓ −2.2
	Fishr	85.6 ± 0.4	78.7 ± 1.1	96.3 ± 0.0	78.9 ± 1.3	84.9 ↓ −0.2
	ERM	81.7 ± 0.9	80.9 ± 0.1	97.1 ± 0.3	80.7 ± 1.0	85.1
	SAGM_DG	**86.1 ± 1.2**	80.7 ± 0.6	96.7 ± 0.1	**83.8 ± 0.5**	86.8 ↑ +1.7
	DFP(Ours)	84.9 ± 0.3	**82.0 ± 1.1**	**97.6 ± 0.2**	80.4 ± 0.7	86.2 ↑ +1.1
In-domain	SWAD	97.6 ± 0.1	97.0 ± 0.3	96.3 ± 0.9	95.6 ± 2.1	96.6 ↓ −0.6
	Fishr	97.9 ± 0.1	97.1 ± 0.1	96.4 ± 0.1	97.8 ± 0.2	97.3 ↑ +0.6
	ERM	97.2 ± 0.0	97.2 ± 0.2	96.4 ± 0.1	98.0 ± 0.1	97.2
	SAGM_DG	98.4 ± 0.1	97.9 ± 0.4	97.3 ± 0.1	98.5 ± 0.0	98.0 ↑ +0.8
	DFP(Ours)	96.6 ± 0.2	97.4 ± 0.3	96.1 ± 0.2	97.8 ± 0.2	97.0 ↓ −0.2

To further understand how DFP affects the performance of the model, we compare the in-domain (the training data and the test data are independent and identically distributed) test accuracies of different methods. As shown in Table 2, DFP has a negative but marginal effect on the in-domain accuracy, suggesting that DFP is able to trade a slightly worse in-domain performance for significantly better out-domain performance.

4.2.2 OfficeHome

Similar to PACS, the OfficeHome dataset also has four training domain combinations, as well as four test domains $\{A, C, P, R\}$. We set $\epsilon \in \{0.1, 0.01, 0.001\}$ for this dataset. As shown in Table 3, DFP improves the average test accuracy by 1.4% over ERM and 1.2% over RSC. While there is a notable improvement on every domain, the overall difficulty of this dataset is substantially higher than that of PACS as indicated by the average accuracies.

Table 3. Test accuracies of ERM, RSC and DFP on OfficeHome.

OfficeHome	A	C	P	R	Avg
ERM	54.8 ± 0.2	49.8 ± 0.4	72.3 ± 0.4	73.4 ± 0.1	62.6
RSC	56.3 ± 0.4	49.5 ± 0.2	72.1 ± 0.1	73.4 ± 0.4	62.8 ↑ +0.2
DFP (Ours)	**57.4 ± 0.9**	**51.0 ± 0.4**	**73.4 ± 0.1**	**74.3 ± 0.4**	**64.0** ↑ +1.4

4.2.3 Terra Incognita

The Terra Incognita dataset also includes four training domain combinations, as well as four test domain types $\{L100, L38, L43, L46\}$. And we set the combination of ϵ and the loss weights $(\epsilon, \alpha, 1-\alpha) \in$ {(0.1,0.9,0.1), (0.01,0.9,0.1), (0.01,0.99,0.01)} for the Terra Incognita dataset. As shown in Table 4, our method can enhance the test accuracy of the Terra Incognita dataset by 1.9% when compared to the ERM. The Terra Incognita dataset contains images of wild animals captured by camera traps in a variety of natural environments,

simulating a real-world scenario. Despite the fact that DFP has improved test accuracies across domains, the baseline and DFP results are not particularly impressive. These results show that OOD generalization is more difficult in photos with more intricate and realistic backgrounds, such as The Terra Incognita dataset.

Table 4. Test accuracies of ERM, RSC and DFP on Terra Incognita.

TerraIncognita	L100	L38	L43	L46	Avg
ERM	44.4 ± 4.2	38.0 ± 2.8	50.5 ± 1.1	36.3 ± 0.1	42.3
RSC	48.4 ± 1.9	40.1 ± 1.8	53.5 ± 0.9	35.8 ± 0.9	44.4 ↑ +2.1
DFP (Ours)	**47.4 ± 1.9**	**39.7 ± 2.7**	**52.6 ± 0.0**	**37.2 ± 0.9**	**44.2** ↑ +1.9

4.2.4 Comparison with other DG methods

Drawing on the findings from Ye et al. [46] and Wang et al. [40], we conduct a comparative analysis between our method and some state-of-the-art techniques. Table 5 presents these results, which are based on ResNet-18, and Table 6 presents test results of more recent methods based on ResNet-50, indicating that our approach consistently surpasses several established methods. Additionally, Table 5 includes our own results for Empirical Risk Minimization (ERM) [36] and Representation Self-Challenging (RSC) [14]. ERM is a standard baseline for comparisons in domain generalization (DG) challenges. Following benchmark results, we also evaluate RSC, which outperforms many other techniques. RSC employs gradient characteristics to mute feature representations with the highest gradient, compelling the model to rely on other features for predictions. In contrast, our method aims to isolate domain-related features and introduce perturbations to them.

Table 5. Test accuracy comparison with other methods based on ResNet-18. Results of the first block are taken from Ye et al. [46].

Method \ Dataset	PACS	OfficeHome	TerraIncognita	Avg
MLDG	73 ± 0.4	52.4 ± 0.2	27.4 ± 2.0	50.93 ↓ −11.54
GroupDRO	80.4 ± 0.3	63.2 ± 0.2	36.8 ± 1.1	60.13 ↓ −2.34
ANDMask	79.5 ± 0.0	62 ± 0.3	39.8 ± 1.4	60.43 ↓ −2.04
Mixup	79.8 ± 0.6	63.3 ± 0.5	39.8 ± 0.3	60.97 ↓ −1.5
MTL	81.2 ± 0.4	62.9 ± 0.2	38.9 ± 0.6	61.00 ↓ −1.47
DANN	81.1 ± 0.4	62.9 ± 0.6	39.5 ± 0.2	61.17 ↓ −1.3
ARM	81 ± 0.4	63.2 ± 0.2	39.4 ± 0.7	61.20 ↓ −1.27
CORAL	81.6 ± 0.6	63.8 ± 0.3	38.3 ± 0.7	61.23 ↓ −1.24
MMD	81.7 ± 0.2	63.8 ± 0.1	38.3 ± 0.4	61.27 ↓ −1.2
ERDG	80.5 ± 0.5	63 ± 0.4	41.3 ± 1.2	61.60 ↓ −0.87
IGA	80.9 ± 0.4	63.6 ± 0.2	41.3 ±0.8	61.93 ↓ −0.54
VREx	81.8 ± 0.1	63.5 ± 0.1	40.7 ± 0.7	62.00 ↓ −0.47
IRM	81.1 ± 0.3	63 ± 0.2	42 ± 1.8	62.03 ↓ −0.44
SagNet	81.6 ± 0.4	62.7 ± 0.4	42.3 ± 0.7	62.20 ↓ −0.27
ERM	81.5 ± 0.0	63.3 ± 0.2	42.6 ± 0.9	62.47
RSC	82.8 ± 0.4	62.9 ± 0.4	43.6 ± 0.5	63.10 ↑ +0.63
ERM (Our runs)	80.1 ± 0.2	62.6 ± 0.1	42.3 ± 1.1	62.03
RSC (Our runs)	82.0 ± 0.5	62.8 ± 0.1	44.4 ± 0.0	63.27 ↑ +1.27
DFP (Ours)	**83.0 ± 0.7**	**64.0 ± 0.3**	**44.2 ± 0.6**	**63.73** ↑ +1.7

Table 6. Test accuracy comparison with recent methods based on ResNet-50. Results of the first block are taken from Wang et al. [40].

Method \ Dataset	PACS	Difference
CDANN	82.6±0.9	↓-2.9
Mixstyle	85.2±0.3	↓-0.3
RSC	85.2±0.9	↓-0.3
Miro (with CLIP)	85.4±0.4	↓-0.1
ERM	85.5±0.2	0.0
Fish	85.5±0.3	0.0
SelfReg	85.6±0.4	↑+0.1
SAM	85.8±0.2	↑+0.3
GSAM	85.9±0.1	↑+0.4
mDSDI	86.2±0.2	↑+0.7
SAGM	86.6±0.2	↑+1.1
DFP(Ours)	86.2	↑ +0.7

4.3 Domain Generalization On Graph Data

According to Wu et al. [42], the graph is a non-linear data structure consisting of vertices V and edges E. And an input graph $G = (E, X)$ contains an adjacency matrix $E = \{e_{vu}|v, u \in V\}$ and node features $X = \{x_v|v \in V\}$. Apart from these, each node in the graph has a label, which can be represented as a vector $Y = \{y_v|v \in V\}$. G_v is a random variable of ego-graphs whose realization is $G_v = (E_v, X_v)$. Based on this, we can adapt the definition of general OOD problem via instantiating the input as G_v and the target as Y, and then the data generation can be characterized as $p(G, Y|e) = p(G|e)p(Y|G, e)$ where e is a random variable of environments that is a latent variable and impacts data distribution. More formally, the OOD problem can be written as:

$$\min_{f} \max_{e\in\varepsilon} \mathbb{E}_{G\sim p(\boldsymbol{G}|\boldsymbol{e}=e)} \left[\frac{1}{|V|} \sum_{v\in V} \mathbb{E}_{y\sim p(y|\boldsymbol{G_V}=G_v,\boldsymbol{e}=e)} [l\left(f(G_v), y\right)] \right]. \tag{10}$$

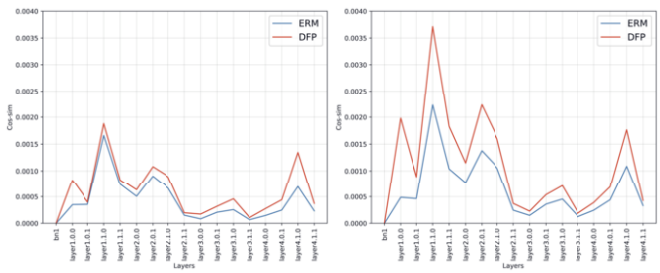
In order to verify the effectiveness of our method when training with multiple graphs and generalizing to unseen domains, we test our method on the FB-100 dataset. We select three groups of data, and each group has three graphs for training, two graphs for validation, and the remaining three for testing. The training graphs are [(John Hopkins, Caltech, Amherst), (Bingham, Duke, Princeton), (WashU, Brandeis, Carnegie)], and we also use test accuracy for evaluation. We use Graph Convolutional Networks (GCN) as the backbone and compare using different configurations of training graphs, as shown in Table 7. We conduct 5 independent runs with the same epochs for each training group. We compare our method with ERM and Explore-to-Extrapolate Risk Minimization (EERM) [42], and the results show that DFP outperforms ERM on average by up to 3.7%. Notably, the test accuracy achieved by our method is comparable to that of EERM, a technique specifically designed to tackle graph OOD issues. These results demonstrate that our method is more adaptable and can be applied to data with various types of structures.

Table 7. Test accuracy on graph data for domain generalization.

Training graphs	Method \ Test	Penn	Brown	Texas	Avg
JohnHopkins	ERM	51.0 ± 0.7	56.4 ± 0.6	55.8 ± 1.2	54.4
+Caltech	EERM	50.2 ± 0.9	56.4 ± 0.4	52.3 ± 1.5	52.9 ↓ −1.5
+Amherst	DFP (Ours)	50.5 ± 0.4	**56.7 ± 0.2**	55.5 ± 1.3	54.2 ↓ −0.2
Bingham	ERM	50.1 ± 1.2	49.7 ± 5.0	50.3 ± 5.9	50.1
+Duke	EERM	50.4 ± 0.8	51.1 ± 4.2	53.1 ± 4.3	51.5 ↑ +1.4
+Princeton	DFP (Ours)	**50.6 ± 1.0**	49.9 ± 2.9	**53.6 ± 3.6**	51.4 ↑ +1.3
WashU	ERM	49.5 ± 1.0	47.2‘± 5.5	54.2 ± 3.9	50.3
+Brandeis	EERM	52.5 ± 1.4	54.1 ± 5.8	56.0 ± 0.5	54.2 ↑ +3.9
+Carnegie	DFP (Ours)	51.7 ± 1.1	**54.3 ± 5.8**	**56.0 ± 0.6**	54 ↑ +3.7

4.4 Gradient Similarity

To illustrate the effect of domain feature perturbation, we examine the similarity computation on the PACS dataset. We define four training domain combinations {(C, P, S), (A, P, S), (A, C, S), (A, C, P)} and their respective test domains $\{A, C, P, S\}$. We employ the gradients' absolute values to compute the cosine similarity across different dimensions of hidden representations. Gradient similarities across domains may reflect feature similarities they emphasize. For each training domain set, we independently assess the similarity. For the three domains in each set, we determine the average gradient of hidden representations as described in **Section 3.3**.



(a) $D_{tr} = \{C, P, S\}$ (b) $D_{tr} = \{A, P, S\}$

(c) $D_{tr} = \{A, C, S\}$ (d) $D_{tr} = \{A, C, P\}$

Figure 5. Gradient similarities at different layers of ERM and DFP.

As illustrated in Figure 5, we calculate the average gradient similarity for each layer subjected to perturbation. To contrast the gradient similarities between DFP and ERM, we select the optimal pretrained model from each method and perform an additional training step. Evaluating each approach 50 times, we determine the mean similarity value. The findings suggest that DFP can induce a similar gradient change between different source domains.

4.5 Ablation Studies

In this section, we present two ablation studies focusing on: (1) a comparison with random perturbations and (2) the optimal position for noise injection. All experiments utilize the ResNet-18 architecture. For both studies, trials are conducted on the PACS dataset with a fixed initial learning rate of lr=5e-5 and the same training steps. In each of the three independent training iterations, we perform eight rounds of random hyperparameter search.

4.5.1 Random perturbation

To evaluate the effectiveness of noise modulation, we train a baseline model with random noise. Experiments using random perturbations are conducted at the same insertion position as DFP. We set the random noise $n \sim \mathcal{N}(0, \sigma^2)$ with $\sigma \in \{0.001, 0.005, 0.01, 0.05, 0.1, 0.2\}$. Table 8 displays the best results of random perturbations. The average accuracy of random perturbations is 81.3%, and the results suggest that our proposed method outperforms random perturbations by 1.7%, which is lower than DFP.

Table 8. Test accuracies of ERM with random perturbations.

PACS	A	C	P	S	Avg
ERM	77.9 ± 0.9	72.7 ± 0.3	95.8 ± 0.2	74.1 ± 0.4	80.1
Random	79.6 ± 0.5	75.3 ± 0.9	95.4 ± 0.2	74.8 ± 0.7	81.3 ↑ +1.2
RSC	79.8 ± 0.2	77.1 ± 0.7	94.8 ± 0.1	76.3 ± 0.6	82.0 ↑ +1.9
DFP (Ours)	**80.6 ± 0.9**	**75.8 ± 1.4**	**96.2 ± 0.3**	**79.3 ± 1.0**	**83.0** ↑ +2.9

4.5.2 Noise injection point

We compare the effect of two different noise injection points, pre-activation and activation. We set the $\epsilon \in \{0.001, 0.005, 0.01, 0.05, 0.1, 0.2\}$ and the loss weights $(\alpha, 1-\alpha) \in \{(0.9,0.1), (0.99,0.01)\}$. Table 9 shows the best results, and there is no notable difference between the two cases.

Table 9. Test accuracies of DFP with different injection points.

PACS	A	C	P	S	Avg
Pre-act	80.6 ± 0.9	74.7 ± 0.6	95.9 ± 0.4	77.7 ± 0.8	82.2
After-act	79.9 ± 0.4	76.0 ± 1.0	96.0 ± 0.3	77.2 ± 0.9	82.2

5 Conclusion

In this work, we proposed a novel approach to domain generalization, named domain feature perturbation (DFP). DFP incorporates a domain classifier to produce perturbations for domain-specific features, aiming to reduce the dependence of the model on such features. Furthermore, we conducted extensive experiments on multiple domain generalization datasets, demonstrating the effectiveness of DFP, as well as competitive or better OOD performance than state-of-the-art methods.

Our method has several limitations. First, DFP relies on two forward and backward passes in each training iteration, rendering it slower at training time than simple approaches such as empirical risk minimization. Second, using the gradient of the domain classifier, we can only approximately identify domain-specific features, making the regularization effect of DFP less precise. Therefore, future research might present methods for the more efficient and precise extraction of these features.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (Grant No. 6220072338).

References

[1] I. Adjabi, A. Ouahabi, A. Benzaoui, and A. Taleb-Ahmed. Past, present, and future of face recognition: A review. *Electronics*, 9(8):1188, 2020.

[2] D. Arpit, H. Wang, Y. Zhou, and C. Xiong. Ensemble of averages: Improving model selection and boosting performance in domain generalization. *Advances in Neural Information Processing Systems*, 35: 8265–8277, 2022.

[3] S. Beery, G. Van Horn, and P. Perona. Recognition in terra incognita. In *Proceedings of the European conference on computer vision (ECCV)*, pages 456–473, 2018.

[4] G. Blanchard, A. A. Deshmukh, Ü. Dogan, G. Lee, and C. Scott. Domain generalization by marginal transfer learning. *The Journal of Machine Learning Research*, 22(1):46–100, 2021.

[5] M.-H. Bui, T. Tran, A. Tran, and D. Phung. Exploiting domain-specific features to enhance domain generalization. *Advances in Neural Information Processing Systems*, 34:21189–21201, 2021.

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

[7] C. Chen, J. Li, X. Han, X. Liu, and Y. Yu. Compound domain generalization via meta-knowledge encoding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7119–7129, 2022.

[8] Q. Fan, M. Segu, Y.-W. Tai, F. Yu, C.-K. Tang, B. Schiele, and D. Dai. Normalization perturbation: A simple domain generalization method for real-world domain shifts. *arXiv preprint arXiv:2211.04393*, 2022.

[9] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky. Domain-adversarial training of neural networks. *The journal of machine learning research*, 17(1): 2096–2030, 2016.

[10] R. Golden. *Statistical machine learning: A unified framework*. CRC Press, 2020.

[11] I. Gulrajani and D. Lopez-Paz. In search of lost domain generalization. *arXiv preprint arXiv:2007.01434*, 2020.

[12] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[13] J. Huang, D. Guan, A. Xiao, and S. Lu. Fsdr: Frequency space domain randomization for domain generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6891–6902, 2021.

[14] Z. Huang, H. Wang, E. P. Xing, and D. Huang. Self-challenging improves cross-domain generalization. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*, pages 124–140. Springer, 2020.

[15] D. Kim, Y. Yoo, S. Park, J. Kim, and J. Lee. Selfreg: Self-supervised contrastive regularization for domain generalization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9619–9628, 2021.

[16] S. Lee, D. Kim, and J. Park. Domain-agnostic question-answering with adversarial training. *arXiv preprint arXiv:1910.09342*, 2019.

[17] D. Li, Y. Yang, Y.-Z. Song, and T. M. Hospedales. Deeper, broader and artier domain generalization. In *Proceedings of the IEEE international conference on computer vision*, pages 5542–5550, 2017.

[18] P. Li, D. Li, W. Li, S. Gong, Y. Fu, and T. M. Hospedales. A simple feature augmentation for domain generalization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8886–8895, 2021.

[19] Z. Li, K. Ren, X. Jiang, Y. Shen, H. Zhang, and D. Li. Simple: Specialized model-sample matching for domain generalization. In *The Eleventh International Conference on Learning Representations*, 2022.

[20] D. Lim, X. Li, F. Hohne, and S.-N. Lim. New benchmarks for learning on non-homophilous graphs. *arXiv preprint arXiv:2104.01404*, 2021.

[21] C. Liu, X. Sun, J. Wang, H. Tang, T. Li, T. Qin, W. Chen, and T.-Y. Liu. Learning causal semantic representation for out-of-distribution prediction. *Advances in Neural Information Processing Systems*, 34:6155–6170, 2021.

[22] F. Lv, J. Liang, S. Li, B. Zang, C. H. Liu, Z. Wang, and D. Liu. Causality inspired representation learning for domain generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8046–8056, 2022.

[23] F. Lv, J. Liang, S. Li, J. Zhang, and D. Liu. Improving generalization with domain convex game. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 24315–24324, 2023.

[24] D. Mahajan, S. Tople, and A. Sharma. Domain generalization using causal matching. In *International Conference on Machine Learning*, pages 7313–7324. PMLR, 2021.

[25] H. Nam, H. Lee, J. Park, W. Yoon, and D. Yoo. Reducing domain gap via style-agnostic networks. *arXiv preprint arXiv:1910.11645*, 2(7):8, 2019.

[26] G. Parascandolo, A. Neitz, A. Orvieto, L. Gresele, and B. Schölkopf. Learning explanations that are hard to vary. *arXiv preprint arXiv:2009.00329*, 2020.

[27] A. Rame, C. Dancette, and M. Cord. Fishr: Invariant gradient variances for out-of-distribution generalization. In *International Conference on Machine Learning*, pages 18347–18377. PMLR, 2022.

[28] S. Sagawa, P. W. Koh, T. B. Hashimoto, and P. Liang. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. *arXiv preprint arXiv:1911.08731*, 2019.

[29] S. Sankaranarayanan, A. Jain, R. Chellappa, and S. N. Lim. Regularizing deep networks using efficient layerwise adversarial training. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[30] M. Segu, A. Tonioni, and F. Tombari. Batch normalization embeddings for deep domain generalization. *Pattern Recognition*, 135:109115, 2023.

[31] Y. Shi, J. Seely, P. H. Torr, N. Siddharth, A. Hannun, N. Usunier, and G. Synnaeve. Gradient matching for domain generalization. *arXiv preprint arXiv:2104.09937*, 2021.

[32] P. P. Shinde and S. Shah. A review of machine learning and deep learning applications. In *2018 Fourth international conference on computing communication control and automation (ICCUBEA)*, pages 1–6. IEEE, 2018.

[33] C. Shorten and T. M. Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of big data*, 6(1):1–48, 2019.

[34] P. Sun, H. Kretzschmar, X. Dotiwalla, A. Chouard, V. Patnaik, P. Tsui, J. Guo, Y. Zhou, Y. Chai, B. Caine, et al. Scalability in perception for autonomous driving: Waymo open dataset. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2446–2454, 2020.

[35] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 23–30. IEEE, 2017.

[36] V. Vapnik. *The nature of statistical learning theory*. Springer science & business media, 1999.

[37] H. Venkateswara, J. Eusebio, S. Chakraborty, and S. Panchanathan. Deep hashing network for unsupervised domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5018–5027, 2017.

[38] Y. Wald, A. Feder, D. Greenfeld, and U. Shalit. On calibration and out-of-domain generalization. *Advances in neural information processing systems*, 34:2215–2227, 2021.

[39] C. Wang, Z. Zhang, and Z. Zhidan. Code and data for "domain feature perturbation for domain generalization", 2024. URL https://github.com/Cguang7/Domain-Feature-Perturbation-for-Domain-Generalization.

[40] P. Wang, Z. Zhang, Z. Lei, and L. Zhang. Sharpness-aware gradient matching for domain generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3769–3778, 2023.

[41] Q. Wang, Y. Wang, H. Zhu, and Y. Wang. Improving out-of-distribution generalization by adversarial training with structured priors. *arXiv preprint arXiv:2210.06807*, 2022.

[42] Q. Wu, H. Zhang, J. Yan, and D. Wipf. Handling distribution shifts on graphs: An invariance perspective. *arXiv preprint arXiv:2202.02466*, 2022.

[43] Y. Wu, Y. Wo, C. Li, and G. Han. Learning domain-invariant representation for generalizing face forgery detection. *Computers & Security*, 130:103280, 2023.

[44] Q. Xu, R. Zhang, Y. Zhang, Y. Wang, and Q. Tian. A fourier-based framework for domain generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14383–14392, 2021.

[45] S. Yan, H. Song, N. Li, L. Zou, and L. Ren. Improve unsupervised domain adaptation with mixup training. *arXiv preprint arXiv:2001.00677*, 2020.

[46] N. Ye, K. Li, H. Bai, R. Yu, L. Hong, F. Zhou, Z. Li, and J. Zhu. Ood-bench: Quantifying and understanding two dimensions of out-of-distribution generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7947–7958, 2022.

[47] M. Yi, L. Hou, J. Sun, L. Shang, X. Jiang, Q. Liu, and Z. Ma. Improved ood generalization via adversarial training and pretraing. In *International Conference on Machine Learning*, pages 11987–11997. PMLR, 2021.

[48] Z. You, J. Ye, K. Li, Z. Xu, and P. Wang. Adversarial noise layer: Regularize neural network by adding noise. In *2019 IEEE International Conference on Image Processing (ICIP)*, pages 909–913. IEEE, 2019.

[49] H. Zhang, Y.-F. Zhang, W. Liu, A. Weller, B. Schölkopf, and E. P. Xing. Towards principled disentanglement for domain generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8024–8034, 2022.

[50] M. Zhang, H. Marklund, A. Gupta, S. Levine, and C. Finn. Adaptive risk minimization: A meta-learning approach for tackling group shift. *arXiv preprint arXiv:2007.02931*, 8:9, 2020.

[51] K. Zhou, Y. Yang, Y. Qiao, and T. Xiang. Domain generalization with mixstyle. *arXiv preprint arXiv:2104.02008*, 2021.

[52] K. Zhou, Z. Liu, Y. Qiao, T. Xiang, and C. C. Loy. Domain generalization: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

[53] W. Zhu, L. Lu, J. Xiao, M. Han, J. Luo, and A. P. Harrison. Localized adversarial domain generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7108–7118, 2022.

[54] J. Zhuang, B. Gong, L. Yuan, Y. Cui, H. Adam, N. Dvornek, S. Tatikonda, J. Duncan, and T. Liu. Surrogate gap minimization improves sharpness-aware training. *arXiv preprint arXiv:2203.08065*, 2022.