A Simple Feature Augmentation for Domain Generalization

Pan Li[1*], Da Li[2,3*], Wei Li[1]
Shaogang Gong[1], Yanwei Fu[4†] and Timothy M. Hospedales[2,3]

[1]Queen Mary University of London [2]Samsung AI Center, Cambridge
[3]University of Edinburgh [4]Fudan University

{pan.li, wei.li, s.gong}@qmul.ac.uk
dali.academic@gmail.com, yanweifu@fudan.edu.cn, t.hospedales@ed.ac.uk

Abstract

The topical domain generalization (DG) problem asks trained models to perform well on an unseen target domain with different data statistics from the source training domains. In computer vision, data augmentation has proven one of the most effective ways of better exploiting the source data to improve domain generalization. However, existing approaches primarily rely on image-space data augmentation, which requires careful augmentation design, and provides limited diversity of augmented data. We argue that feature augmentation is a more promising direction for DG. We find that an extremely simple technique of perturbing the feature embedding with Gaussian noise during training leads to a classifier with domain-generalization performance comparable to existing state of the art. To model more meaningful statistics reflective of cross-domain variability, we further estimate the full class-conditional feature covariance matrix iteratively during training. Subsequent joint stochastic feature augmentation provides an effective domain randomization method, perturbing features in the directions of intra-class/cross-domain variability. We verify our proposed method on three standard domain generalization benchmarks, Digit-DG, VLCS and PACS, and show it is outperforming or comparable to the state of the art in all setups, together with experimental analysis to illustrate how our method works towards training a robust generalisable model.

1. Introduction

Deep learning methods demonstrate exceptional performance in different fields of computer vision, such as object recognition, semantic segmentation or object detection. However, these machine learning systems' performance drops dramatically when encountering test data, which is statistically different from the training data [5]. This issue is known as the domain shift problem, which domain generalization (DG) research aims to address. Models with good DG properties are crucial in practical applications since the distribution of testing data in deployment is inevitably different from training data collected for model fitting [17], whether due to either the expense or simple impossibility of collecting representative training data.

DG research traces back to a decade ago [3]. Since then a variety of methods were proposed to push the DG boundary, including learning domain-invariant features [28, 14], extracting the underlying domain knowledge [15, 20], and

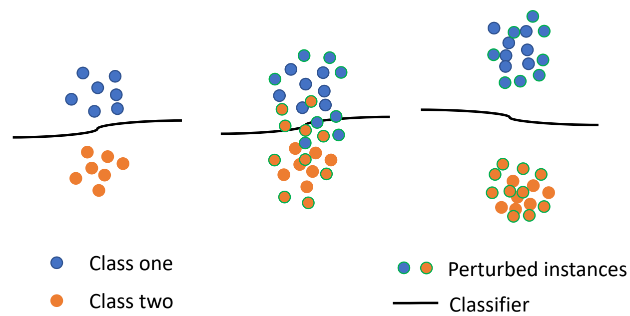

Figure 1: Illustrative schematic of our stochastic feature augmentation method. The trained vanilla model has limited robustness as simple perturbed feature instances, as might be encountered experiencing domain-shift, could induce the classifier to make a mistake. During the training, we persistently perturb the feature embedding, which leads to classification mistakes. In order to discriminate these erroneous perturbed instances, the feature spare must adapt to separate the classes with a more robust decision boundary. This new boundary is in turn more robust to domain-shift.

∗ Equal contributions.
† Corresponding author.

meta-learning inspired methods [21, 2, 6, 22]. Among existing DG strategies, data augmentation based approaches [33, 38, 44] have become popular. Data augmentation is already widely used to reduce overfitting in conventional supervised learning [18], by inserting predefined class-preserving operations, such as transformation, cropping, rotation, flipping. Intuitively, augmenting the source domain data with diverse samples better representing the breadth of plausible domains also leads to improved generalization to novel domains, especially when there are only a few known source domains to start with. However, existing augmentation-based approaches primarily rely on image-space augmentations, which are non-trivial to design due to the difficulty of specifying- or learning how to synthesize images in new domains. Existing approaches include perturbing inputs by gradient-descent on the signal from a domain classifier [33], generating adversarial samples [38] and using an image synthesis network to generate novel images that fool a domain classifier [44]. These approaches are all computationally expensive and complex. In contrast, feature-level data augmentation have also proven effective recently in a supervised learning context [37]. We are inspired by these ideas to explore feature-level augmentation solutions to DG, as shown in Fig. 1.

In this paper, we first show that an extremely simple feature augmentation of perturbing latent features using white Gaussian noise already leads to comparable performance to recent state-of-the-art. This strategy outperforms the above mentioned highly engineered approaches that rely on training image-to-image generation networks, or gradient-based adversarial sample generation; while being extremely simple to implement and much faster to run. Nevertheless, while feature augmentation helps to enhance the breadth of seen domains for training, a limitation is that one can not add too much noise without risking inducing a non-class-preserving augmentation, which then has the counter-productive effect of introducing label-noise. To enable more meaningful and class-preserving augmentations, we aim to estimate the natural directions of correlation that already exist in a given source dataset. Specifically, we estimate the feature covariance online during training using moving average and then use it to simulate a joint (multivariate Normal) noise distribution across the features. Estimating class-conditional covariance further ensures that the learned noise follows the class-preserving but inter-domain directions.

The proposed method potentially applies to any base DG method. We show that it improves the vanilla method to achieving the state of the art performance on three benchmarks, Digit-DG, VLCS and PACS. Furthermore, we show the feature evolution from pretrained Vanilla to our SFA trained features to understand how the proposed method essentially improves model generalization.

2. Related Work

2.1. Domain Generalization

Various methods have been proposed in the DG literature, spanning shallow [15, 28, 14] and deep [20, 27, 21, 2, 6, 39, 44, 46] learning methods. Representative works can be categorized into: domain invariant/agnostic model learning [28, 14, 27, 23, 15, 20], meta-learning based DG methods [21, 2, 6, 22], data augmentation based DG methods [33, 38, 44], and self-supervision based methods [4, 39].

Our work is most relevant to the data augmentation based DG methods. [33] designed a special Bayesian architecture and required one-step of back-propagation to generate image perturbations. [38] targeted finding the 'hardest' (adversarial) samples for the current model and appending them into the training data, which requires a costly minimax optimization at each iteration. [44] proposed to generate novel augmentation images via domain-adversarial training of an image generation network, extended with optimal-transport based metrics in [45]. In this paper, we take a different perspective and propose a simple yet effective feature-level data augmentation technique for DG. Our feature augmentation could be implemented with a few lines of code and applies to any base DG methods in a plug-in manner.

2.2. Feature Interpolation and Augmentation

It has been observed that deep features are usually well linearized [36], so that simple vector interpolation can change the content of a feature [13]. [37] also demonstrated that feature augmentation by simply mixing up features improves model generalization. Our approach is inspired by such feature augmentation techniques. However, unlike existing works [13, 36] which learn the interpolation for some 'semantic' change, our feature augmentation aims to retain category information and span the space of domains.

Unlike some recent work [46], which explicitly perturbs latent features in a mix-up inspired way [37] by exploiting domain labels, our method perturbs feature embedding without using any domain labels.

2.3. Domain Randomization

Domain randomization has been widely used in different tasks [34, 43, 31, 31, 42, 26] in computer vision. [34] firstly proposed to apply diverse random rendering styles to synthetic data, such that the test data was likely to lie within the training distribution, thus improving generalization. [31] proposed to improve domain randomization by the guidance from the task-specific prior knowledge. Our proposed data adaptive noise generator preserves the intra-class variations, but fits to the inter-domain variation observed among training domains. Thus, when using the sampled noise to perturb the original feature, it interpolates the features along directions spanning the inter-domain directions while preserving inter-class directions, thus serving as an implicit domain randomization.

2.4. Stochastic Neural Networks

Incorporating random variables during learning and inference is common in variational inference models, such as Varitional Autoencoder (VAE) [16] and Variational Information Bottleneck (VIB) [1], to fulfill the optimization of their KL divergence term. Recently, some researchers [25, 41] also observed the benefits of injecting noise into the latent feature representation for improving adversarial defense. However, all their focuses are different to us and none of them has considered modeling feature correlation in their noise sampling.

3. Methodology

Problem Definition. In the typical domain generalization setting, there are N multiple source domains, $\{\mathcal{D}^1, \ldots, \mathcal{D}^N\}$, where each $\mathcal{D}^j$ contains M (M may vary across different domains) data and label pairs with joint distribution $P^j(\mathcal{X}, \mathcal{Y}) : \mathbf{x} \in \mathcal{X}, y \in \mathcal{Y}\}_{j=1}^{M}$. The goal in DG is to train a model on the source domains, and test it on an unseen target domain $\mathcal{D}^{N+1}$. The unseen target domain is normally different from the source domains as $P^j(\mathcal{X}, \mathcal{Y}) \neq P^{N+1}(\mathcal{X}, \mathcal{Y}), j = 1, \ldots, N$.

3.1. Vanilla ERM Baseline

Let us assume the training model, namely a deep convolutional neural network, consists of a feature extractor F_Θ and a classifier G_ψ, parameterized by Θ and ψ respectively. Typically, the feature extractor F_Θ is formed of multiple separate layers $F^i_{\Theta^i}$. Then, given an input image $\mathbf{x}$, its prediction through this model is $\hat{y} = G_\psi \circ F_\Theta(\mathbf{x}) = G_\psi \circ F^L_{\Theta^L} \circ \cdots \circ F^1_{\Theta^1}(\mathbf{x})$, where L means the number of total layers of the feature extractor. At training, a vanilla baseline is trained using source domain data by the empirical risk minimization (ERM) objective.

$$\begin{aligned} &\arg\min_{\Theta,\psi} \; L_{\text{erm}}(\{\mathcal{D}^j\}, F_\Theta, G_\psi) \\ \Longrightarrow &\arg\min_{\Theta,\psi} \; \frac{1}{N}\sum_{j=1}^{N} \frac{1}{|\mathcal{D}^j|} \sum_{\mathbf{x},y\in\mathcal{D}^j} \ell(G_\psi \circ F_\Theta(\mathbf{x}), y) \end{aligned} \tag{1}$$

where ℓ is the loss function, i.e. cross entropy loss in our problem. We build on vanilla ERM here, as it is known as a very strong baseline [20, 17]. Neveretheless, one can apply our SFA on top of almost any alternative base DG method.

3.2. Feature Augmentation

Some previous works [35, 37, 46] have suggested that various kinds of feature augmentation can help to improve model generalization. Different from the standard data augmentation on the raw image space, feature augmentation applies the 'transformation' directly on the feature space. For instance, given a latent embedding tensor $\mathbf{z}^i = F^i(\mathbf{z}^{i-1})$, $\mathbf{z}^i \in \mathbb{R}^{N\times K\times H\times W}$ as the output of the layer i of the feature extractor, it can be augmented as

$$\hat{\mathbf{z}}^i = A(\mathbf{z}^i) \tag{2}$$

where $A(.)$ is an feature augmentation function, which will perturb the vanilla representation $\mathbf{z}^i$ into a new representation $\hat{\mathbf{z}}^i$. For one instance, in [37] A is implemented as,

$$A(\mathbf{z}^i) = a * \mathbf{z}^i|_{y=c1} + (1-a) * \mathbf{z}^i|_{y=c2} \tag{3}$$

which mixes up two features from two different classes to a novel interpolated vector.

3.3. Stochastic Feature Augmentation (SFA)

In this paper, we propose to augment feature representation using random noise. More specifically, the latent feature embedding is augmented by simply multiplying and adding random variables sampled from certain distributions. The feature augmentation function is formulated as

$$\hat{\mathbf{z}}^i = A(\mathbf{z}^i) = \boldsymbol{\alpha} \odot \mathbf{z}^i + \boldsymbol{\beta} \tag{4}$$

where $\boldsymbol{\alpha} \in \mathbb{R}^{N\times K\times H\times W}$ and $\boldsymbol{\beta} \in \mathbb{R}^{N\times K\times H\times W}$ are the noise samples, and $\odot$ indicates element-wise multiplication. Each element $\boldsymbol{\alpha}$ is sampled from some distributions, i.e. Normal distributions $\mathcal{N}(\mu, \Sigma)$ in this paper. In particular, the statistical moments (μ, Σ) can be hyperparameters or updated during the model optimization by incorporating features' statistics. In the following we will introduce two variants of our SFA, including a simple white noise baseline and an adaptive full-covariance version.

3.3.1 Simple Data-independent Noise

First we investigate the simplest version of this stochastic feature augmentation. We set μ as constant and Σ as a constant diagonal matrix, formulating SFA as a data-independent stochastic feature augmentation module. With this parameterisation, the SFA function can be treated as perturbing the original feature representations randomly without preferring any particular perturbation directions. Now the scale α and bias β are sampled from two multivarate Gaussian distributions,

$$\begin{aligned} \boldsymbol{\alpha} &\sim \mathcal{N}(1, \sigma_1 \boldsymbol{I}) \\ \boldsymbol{\beta} &\sim \mathcal{N}(0, \sigma_2 \boldsymbol{I}) \end{aligned} \tag{5}$$

where, $\boldsymbol{I}$ is the identity matrix, σ_1, σ_2 are two scalar hyperparmeters. By default we set $\sigma_1 = \sigma_2$ to reduce hyperparameters. Although the perturbation does not follow any 'meaningful' direction here, it empirically already improves the base ERM method significantly.

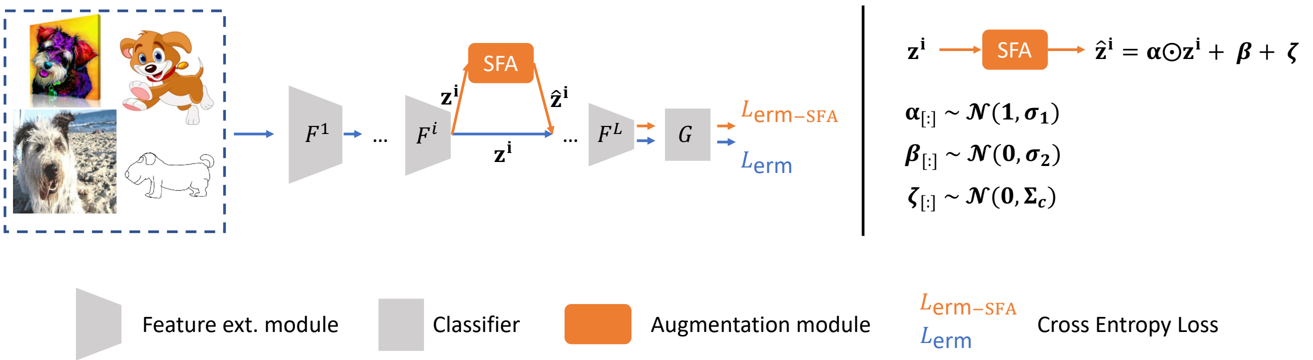

Figure 2: The overall pipeline of our method. Our SFA module is a plug-in module which can be incorporated into any neural DG method. As illustrated on the right, our SFA perturbs the latent features using a combination of non-adaptive uniform and data-adaptive shaped noise.

3.3.2 Adaptive Dependent Noise

The simple strategy above samples the random noise from a data-independent uniform diagonal Normal distribution. As we will show later, this is already a highly effective regularizer for DG. Nevertheless, an issue is that the sampled noise does not follow the class-preserving directions, so while there are augmentation benefits, these could be undermined in practice by effectively introducing label-noise. Second, sampling uniform-diagonal noise to perturb the feature embedding, does not consider the correlation between different feature dimensions which can be important for the semantics of the representation. In particular, we would like to sample more aggressively along latent directions spanned by the data domains; and less aggressively along latent dimensions spanned by different categories.

We therefore propose an adaptive data dependent regularizer in which we assume that, besides the uniform noise generator above, we also have a class-conditional noise generator with its own full covariance matrix. Now the SFA function is defined as

$$\begin{aligned} \hat{\mathbf{z}}^i &= A(\mathbf{z}^i) \\ &= \boldsymbol{\alpha} \odot \mathbf{z}^i + \boldsymbol{\beta} + \boldsymbol{\xi} \end{aligned} \tag{6}$$

where $\boldsymbol{\xi} \in \mathbb{R}^{N \times K}$ is sampled from class-specific Multivariate Normal distributions $\mathcal{N}(0, \boldsymbol{\Sigma}_c), c \in [1, C], \boldsymbol{\Sigma}_c \in \mathbb{R}^{K \times K}$.

Next we specify how to estimate these class-wise covariances Σ_c. Our strategy is to fit them to the observed data covariance. Since there is insufficient data in each mini-batch to do this accurately, we intermittently estimate the covariance using data from K mini-batches and update them online using exponential moving average (EMA) as

$$\boldsymbol{\Sigma}_c = \lambda * \boldsymbol{\Sigma}_c + (1 - \lambda) * \mathrm{Cov}(\mathrm{Mean}_{\mathrm{dim}=(2,3)}(\mathbf{z}^i)[y = c]) \tag{7}$$

where λ is the discount factor.

Algorithm 1: Stochastic Feature Augmentation (SFA)

Data: feature $\mathbf{z}^i$; label y; noise scalars σ_1, σ_2; EMA rate λ; bool $adaptive$.
Result: augmented feature $\hat{\mathbf{z}}^i$.

```
/* SFA-S: data-independent noise */
```
Sample $\boldsymbol{\alpha} \sim \mathcal{N}(1, \sigma_1 \boldsymbol{I})$ and $\boldsymbol{\beta} \sim \mathcal{N}(0, \sigma_2 \boldsymbol{I})$;
Compute the augmented feature $\hat{\mathbf{z}}^i = \boldsymbol{\alpha} \odot \mathbf{z}^i + \boldsymbol{\beta}$;
```
/* SFA-A: adaptive noise */
```
if $adaptive == True$ **then**
  Estimate the class-wise covariances $\boldsymbol{\Sigma}_c = \lambda * \boldsymbol{\Sigma}_c + (1 - \lambda) * \mathrm{Cov}(\mathrm{Mean}_{\mathrm{dim}=(2,3)}(\hat{\mathbf{z}}^i)[y = c])$;
  Sample $\boldsymbol{\xi}[y = c, :] \sim \mathcal{N}(0, \boldsymbol{\Sigma}_c)$;
  Compute the augmented feature $\hat{\mathbf{z}}^i = \hat{\mathbf{z}}^i + \boldsymbol{\xi}$;
end

Compared to the baseline in Sec 3.3.1 we now model feature correlations across the hidden dimensions. Note that because the covariance is taken over all instances, aggregating over domains, directions of high inter-domain variability will be modeled by these covariances. However, because we use the labels to fit class-conditional covariances, intra-class variations are preserved in Σ_c.

During the forward pass, elements of $\boldsymbol{\xi}$ are sampled as,

$$\boldsymbol{\xi}[y = c, :] \sim \mathcal{N}(0, \boldsymbol{\Sigma}_c) \tag{8}$$

where the minibatch feature instances from the same class share the same noise.

Loss Function The loss function for using SFA is

$$L_{\text{erm-sfa}} = \frac{1}{N} \sum_{j=1}^{N} \frac{1}{|\mathcal{D}^j|} \sum_{\mathbf{x}, y \in \mathcal{D}^j} \ell(G_\psi \circ \hat{F}_\Theta(\mathbf{x}), y) \tag{9}$$

where $\hat{F}_\Theta(\mathbf{x})$ is the latent feature embedding augmented by our SFA module in the forward pass. Since the multivariate normal distribution in Eq. (8) is straightforward to reparamaterize, back-propagation requires no special machinery.

The overall pipeline of our proposed method is illustrated in Fig. 2. Meanwhile, the pseudo code of our proposed module is in Alg. 1, which show just how easy it is to implement our SFA. Practically, we apply l2 norm on the final feature, stabilizing model training. We checked that removing our SFA module while retaining the l2 norm does not give any improvement, confirming the effectiveness is all from our proposed module.

3.4. Training and Inference

During training we optimize the model parameters using the following objective,

$$\arg\min_{\Theta,\psi} \quad L_{\text{erm}} + \gamma L_{\text{erm-sfa}} \tag{10}$$

At inference, the SFA module will be disabled and feature extractor F_Θ is deterministic. Although we introduce σ_1, σ_2 ($\sigma_1 = \sigma_2$) in our method, we empirically find that they are consistent across all benchmarks and do not require much tuning.

4. Experiments

4.1. Datasets and Settings

Datasets. To evaluate the efficacy of our proposed method, we conduct extensive comparative evaluations on three DG benchmarks: Digit-DG [45], VLCS [8], and PACS [20]. Specifically, Digit-DG is a combination of four handwritten digit recognition datasets (MNIST [19], MNIST-M [11], SVHN [29] and SYN [11]) with domain shit mainly in font style, stroke color and background. VLCS contains five classes (bird, car, chair, dog, and person) which were collected from four photo datasets (PASCAL VOC 2007 [9], LabelMe [32], Caltech [7], and Sun [40] datasets). PACS consist of seven classes of images (dog, elephant, giraffe, guitar, horse, house, and person) for training and testing. These images are depicted in four image styles (photo, art, cartoon, sketch), presenting larger domain discrepancy than the other two. In all experiments, we strictly followed the previous works [39, 44] which used standard image augmentations on PACS&VLCS but not on Digit-DG.

Evaluation Protocol. Following prior works [22, 20, 4], we apply the leave-one-domain-out protocol for fair comparisons. Namely, during the training, one domain is selected as the held out domain while the remaining domains are treated as source domains for model training. For performance measure, we report top-1 classification accuracy on each test domain and the average accuracy accordingly.

Competitors. We evaluate both stochastic feature augmentation methods including the simple non-adaptive variant (SFA-S, Sec 3.3.1), and the adaptive variant (SFA-A, Sec 3.3.2). We compare our methods with fifteen state-of-the-art DG methods categories in Tab. 1, and a baseline method (named Vanilla) that directly aggregates all the source domain data for model training without any DG tricks. As we previously described in Sec. 2.1, these competitors span different types. However, in our evaluation context, we primarily aim to study the efficacy of augmentation for DG. To this end, we summarized these DG methods into two categories: (**A**) augmentation-based DG and (**B**) non-augmentation-based DG. Also, we conducted a rigorous assessment by marking those methods that explicitly leveraged domain label information in model training.

Category	Method	Domain Supervision
B	Vanilla ERM	✗
B	JiGen [4]	✗
B	RSC [30]	✗
B	EISNet [39]	✗
B	MASF [6]	✓
B	CCSA [27]	✓
B	DANN [12]	✓
B	MAML [10]	✓
B	MLDG [21]	✓
B	MetaReg [2]	✓
B	Epi-FCR [22]	✓
B	MMD-AAE [23]	✓
A	CrossGrad [33]	✓
A	DDAIG [44]	✓
A	L2A-OT [45]	✓
A	SFA-S(Ours)	✗
A	SFA-A(Ours)	✗

Table 1: Domain generalization method categorization by augmentation and domain supervision. **A**: augmentation-based DG; **B**: non-augmentation-based DG. ✓: Requires domain labels; ✗: Does not require domain labels.

4.2. Evaluation on Digit-DG

Implementation details. We use exactly the same network configuration in previous works [44], which consists of four Conv (3×3, 64 kernels) – ReLU – Maxpooling (2×2) modules and one softmax classification layer. We warm up the model from scratch first for 500 iterations using only the ERM loss and then add SFA loss for the rest 5500 iterations. We use M-SGD with batch size 42, momentum 0.9, learning rate 0.01 for classifier and 0.01(0.02) for feature extractor in SFA-S(SFA-A) and weight decay 5e-4. During training, our SFA module is added on the penultimate feature layer to perform feature augmentation for source domains. We set $\sigma_1 = \sigma_2 = 1, \gamma = 3.0$. For SFA-A, we update the estimated $\mathbf{\Sigma}_c$ every $K = 8$ batches and set $\lambda = 0.3$.

Results. We summarize the evaluation results on Digit-DG with comparison to the state-of-the-art methods in Tab. 2. Using simple data-independent noise for feature augmentation, our SFA-S model outperforms the existing best augmentation-based DG method (L2A-OT [45]) by a

Target	MNIST	MNIST-M	SVHN	SYN	Ave.
Vanilla ERM	95.8	58.8	61.7	78.6	73.7
CCSA [27]	95.2	58.2	65.5	79.1	74.5
MMD-AAE [23]	96.5	58.4	65.0	78.4	74.6
JiGen [4]	96.5	61.4	63.7	74.0	73.9
CrossGrad [33]	96.7	61.1	65.3	80.2	75.8
DDAIG [44]	96.6	64.1	68.6	81.0	77.6
L2A-OT [45]	**96.7**	63.9	68.6	83.2	78.1
SFA-S(Ours)	**96.7**	66.3	68.8	**85.1**	79.2
SFA-A(Ours)	96.5	**66.5**	**70.3**	85.0	**79.6**

Table 2: Domain Generalization results on **Digit-DG** benchmark.

clear margin of 1.1%. Please note, L2A-OT requires careful design and expensive training cost to achieve these results. In contrast, our SFA-S only requires some trivial noise samples, as described in Alg. 1, and is trained at the same speed as the Vanilla ERM lower bound. Our SFA-A model with adaptive data-dependent noise achieves the best overall performance (Avg.), which further improves the DG performance upon our SFA-S model by 0.4%. Notably, on the most two difficult target domains (MNIST-M and SVHN), our SFA-A model obtains significant improvements (+7.7% and +8.6% respectively) compared with the Vanilla model. This shows that our SFA can effectively deal with large domain shifts caused by complex backgrounds and cluttered digits.

4.3. Evaluation on PACS

Implementation details. We follow the latest state of the art art [39] using ResNet-18 (ImageNet pretrained) as our backbone model and use the predefined protocol in [20] for fair comparison. We train the model with the same training strategy as above for total $3.5k$ iterations and set learning rate 0.001 for classifier and 0.001(0.002) for feature extractor for SFA-S(SFA-A). Again we add our SFA-S/SFA-A in the penultimate feature layer and set $\sigma 1 = \sigma 2 = 1, \gamma = 1.0$. For SFA-A, we update the estimated $\mathbf{\Sigma}_c$ every $K = 12$ batches and set $\lambda = 0.3$.

Results. From the results in Tab. 3, we can see that again our SFA-S variant has already achieved the comparable performance to the previous state of the art methods. Our SFA-A improves the simple variant SFA-S with 0.7% accuracy, resulting in a clear improvement margin of 2.6% accuracy over the vanilla ERM method and outperforming the data augmentation DG method CrossGrad [33] with 3.9%. Although not as good as recent state of the art methods, such as EISNet and L2A-OT [45], our method wins in term of its efficiency and simpleness. And note that L2A-OT uses domain labels at training, whereas we do not.

Target	Art.	Cartoon	Photo	Sketch	Ave.
Vanilla ERM	77.6	73.9	94.4	70.3	79.1
DANN [12]	81.3	73.8	94.0	**74.3**	80.8
MAML [10]	78.3	76.5	95.1	72.6	80.6
MLDG [21]	79.5	77.3	94.3	71.5	80.7
CrossGrad [33]	78.7	73.3	94.0	65.1	77.8
MetaReg [2]	79.5	75.4	94.3	72.2	80.4
Epi-FCR [22]	82.1	77.0	93.9	73.0	81.5
RSC† [30]	78.9	76.9	94.1	76.8	81.7
EISNet [39]	81.9	76.4	95.9	**74.3**	82.2
L2A-OT [45]	**83.3**	**78.2**	**96.2**	73.6	**82.8**
SFA-S(Ours)	80.1	76.2	94.1	73.5	81.0
SFA-A(Ours)	81.2	77.8	93.9	73.7	81.7

Table 3: Domain Generalization evaluation results on **PACS** dataset with **ResNet18**. † is the result reproduced by [30].

4.4. Evaluation on VLCS

Implementation details. We follow the latest work EISNet [39] that uses AlexNet (ImageNet pretrained) on this benchmark. We follow the train/test protocol as per [39]. We set the learning rate 0.0002 for classifier and 0.0001(0.0002) for feature extractor for SFA-S(SFA-A). We train the model using the same strategy as above for total $4k$ iterations. We set $\sigma 1 = \sigma 2 = 1, \gamma = 1.0$ same as above. For SFA-A, we update the estimated $\mathbf{\Sigma}_c$ every $K = 20$ batches and set $\lambda = 0.3$.

Results. From the results in Tab. 4, we can see that our SFA-S achieves the comparable overall performance to all the recent DG competitors except EISNet [39], and stands out when V is the heldout domain, again demonstrating the effectiveness of this simple feature augmentation technique. SFA-A, which shows the best performance heldout C domain, now improves SFA-S with a margin of 0.5% and the Vanilla ERM method 1.3%, demonstrating its consistent effectiveness as the previous setups.

5. Further Analysis

Choice of noise distribution. We chose to use Normal distribution across all benchmarks. We also compared other potential noise distributions including Laplace and uniform in Tab. 5 using SFA-S(β) baseline (additive noise only). From the results we can see that: (i) simple Gaussian noise is best overall, although (ii) any choice of noise distribution provides an consistent improvement on the deterministic ERM baseline.

Where to add SFA module? We explore adding our SFA module at different potential locations within the 4-layer CNN used for Digit-DG. From the results in Tab. 6, we can see that: (i) injecting noise at Layer 2 is the best, but (ii) any location for noise injection provides an improvement on the deterministic Vanilla ERM baseline.

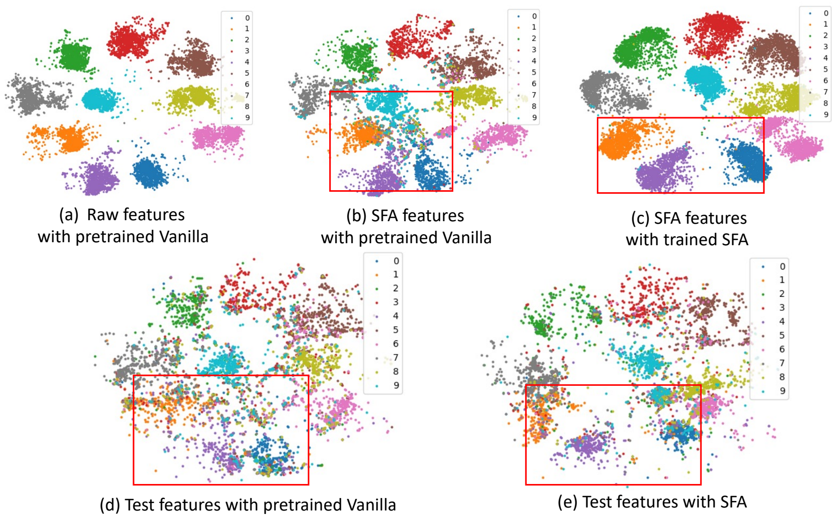

Figure 3: T-SNE visualization of features on **Digit-DG**. Plots (a-c) the features of training data, and plots (d-e) are the features of test data.

Target	V	L	C	S	Ave.
Vanilla ERM	**71.9**	59.2	96.9	62.6	72.7
D-MTAE [14]	63.9	60.1	89.1	61.3	68.6
CIDDG [24]	64.4	63.1	88.8	62.1	69.6
CCSA [27]	67.1	62.1	92.3	59.1	70.2
DBADG [20]	70.0	63.5	93.6	61.3	72.1
MMD-AAE [23]	67.7	62.6	94.4	64.4	72.3
MLDG [21]	67.7	61.3	94.4	65.9	72.3
Epi-FCR [22]	67.1	64.3	94.1	65.9	72.9
JiGen [4]	70.6	60.9	96.9	64.3	73.2
MASF [6]	69.1	**64.9**	94.8	67.6	74.1
EISNet [39]	69.8	63.5	**97.3**	**68.0**	**74.7**
SFA-S(Ours)	69.6	62.2	96.2	65.8	73.5
SFA-A(Ours)	70.4	62.0	97.2	66.2	74.0

Table 4: Domain Generalization evaluation results on **VLCS** dataset with **AlexNet**.

Target	M	MM	SV	SY	Ave.
Vanilla ERM	95.8	58.8	61.7	78.6	73.7
SFA-S(Uniform)	95.6	63.8	62.3	83.0	76.2
SFA-S(Laplace)	95.5	64.3	65.4	82.1	76.8
SFA-S(Normal)	95.9	64.0	65.0	82.9	77.0

Table 5: Analysis of noise type using **SFA-S(β)** on **Digit-DG**.

Parameter sensitivity analysis on noise strength. Our simple model SFA-S has only a single hyperparameter $\sigma_1 = \sigma_2$ of noise strength. In Tab. 7, we perform a sensitivity study on this hyperparameter. From the results we can see that a moderate noise strength of $\sigma_1 = \sigma_2 = 1$ performs best, but SFA is not sensitive to the specific setting of this hyperparameter as it clearly surpasses the Vanilla ERM baseline across six orders of hyperparameter magnitude.

Target	M	MM	SV	SY	Ave.
Vanilla ERM	95.8	58.8	61.7	78.6	73.7
SFA-S(Layer 0)	96.3	64.7	64.2	81.9	76.8
SFA-S(Layer 1)	96.3	66.0	66.9	82.9	78.0
SFA-S(Layer 2)	**96.7**	**66.3**	**68.8**	85.1	**79.2**
SFA-S(Layer 3)	95.8	65.5	66.8	**87.3**	78.9

Table 6: Analysis of location choice to inject noise using **SFA-S**(α, β) on **Digit-DG** benchmark.

Target	M	MM	SV	SY	Ave.
Vanilla ERM	95.8	58.8	61.7	78.6	73.7
SFA-S(0.01)	96.2	65.4	67.9	85.0	78.6
SFA-S(0.1)	96.4	65.9	68.1	85.3	78.9
SFA-S(1)	96.7	66.3	68.8	85.1	79.2
SFA-S(10)	96.4	65.5	67.4	83.8	78.3
SFA-S(100)	96.4	65.2	67.2	84.3	78.3
SFA-S(1000)	96.0	64.5	67.3	84.8	78.2

Table 7: Parameter sensitivity analysis of **SFA-S**(α, β) with different values of $\sigma_1 = \sigma_2$ on **Digit-DG**.

Ablation study of noise components in SFA. We conduct a thorough study of three noise components in our SFA: scale α, bias β, and covariance ξ. In Tab. 8, there are different variants: non-adaptive additive/multiplicative noise (SFA-S(β)/SFA-S(α)), adaptive additive noise (SFA-S(ξ)) and their different combinations (SFA-S(β, α)), (SFA-S(ξ, α))

Target	M	MM	SV	SY	Ave.
Vanilla ERM	95.8	58.8	61.7	78.6	73.7
SFA-S(β)	95.9	64.0	65.0	82.9	77.0
SFA-A(ξ)	95.2	64.6	66.5	82.9	77.3
SFA-S(α)	96.1	65.6	67.5	84.1	78.3
SFA-S(β, α)	**96.7**	66.3	68.8	**85.1**	79.2
SFA-A(ξ, α)	96.0	**67.2**	69.4	84.4	79.3
SFA-A(β, α, ξ)	96.5	66.5	**70.3**	85.0	**79.6**

Table 8: Ablation study of **SFA** noise components on **Digit-DG**.

Target	Art	Cartoon	Photo	Sketch	Ave.
EISNet [39](*)	82.6	75.3	94.9	74.8	81.9
EISNnet+SFA-A	82.1	76.7	95.5	75.7	82.5
DDAIG [44](*)	80.2	75.0	94.4	74.5	81.0
DDAIG+SFA-A	81.3	75.9	94.6	75.2	81.8

Table 9: Results of incorporating SFA on other SoTA methods on **PACS** using **ResNet18**. *: results reproduced by running their public code base.

and (SFA-A(β, α, ξ)). Table 8 shows that: (i) simply using non-adaptive additive noise bias only β can obtain a considerable improvement (+3.3%) compared with the Vanilla ERM method, and using adaptive noise ξ gives larger improvement margin (+3.6%) which is expected. Among all single-noise variants, the multiplicative noise α works most effectively enabling a 4.6% gain. (ii) using the combination of different noise works better than single noise and combining ξ and α is more beneficial than combining β and α, (iii) the full model SFA-A(β, α, ξ) using all different noise further improves the quality of feature augmentation resulting in a 5.9% gain over the Vanilla ERM method.

Incorporating SFA on other base DG methods. Besides incorporating our SFA on top of the Vanilla ERM method, we also investigate its efficacy on improving SoTA DG methods EISNet [39] and DDAIG [44]. We can clearly observe that both base DG methods get improved (+0.6%, +0.8%) by our SFA module on the challenging PACS benchmark using ResNet-18 backbone in Tab. 9. Please note that DDAIG is an image-augmentation based DG method, whereas EISNet is not. This observation further shows that our SFA works well with different DG alternatives and confirms that our method is complemented with image-augmentation techniques.

Visualization of trained features. To illustrate the impact of SFA augmentation on representation learning, Figure 3 provides t-SNE visualization of the penultimate feature layer for the Digit-DG benchmark. The first three plots (a)-(c) are from source domain training data. We can see that while the initial ERM features are well separated (a), applying the SFA module on these (fixed) features without adaptation leads to a blurring of category boundaries and overlapped features (b). When performing representation learning under SFA augmentation (c), the encoder is forced to learn a better and more robust feature separation compared to the initial ERM baseline in order to reduce the overlap despite the injected perturbation, and minimise the loss. The second two plots (d-c) correspond to test data drawn from a novel target domain. Due to the domain shift all the features are less well distributed. However thanks to the additional robustness imbued by the SFA augmentation in (c), the test data remains more separable under domain-shift (compare Figure 3(e) vs (d)).

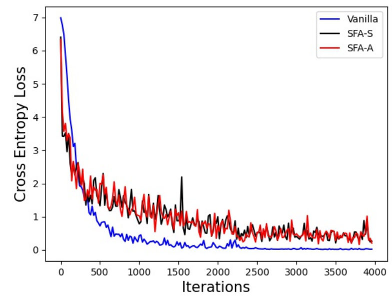

Figure 4: The loss curve of the **Vanilla ERM**, **SFA-S** and **SFA-A** on **Digit-DG**.

Training loss curve. We visualize the training loss curve of the vanilla method with and without incorporating our SFA modules. From the loss curve in Fig. 4, it is seen in both case the training loss is reasonably minimized, indicating that injecting random noise does not break the model convergence. Meanwhile, our SFA-S and SFA-A have slightly larger loss than Vanilla ERM method during optimization, which is expected as our methods try to confuse classifier leading to more mistakes.

6. Conclusion

In this paper we proposed a simple stochastic feature augmentation approach to improving model performance under domain shift. Our SFA provides a simple plug-in module, that provides state of the art performance when used to augment a Vanilla ERM baseline. Unlike alternative data-augmentation based approaches which are extremely complex, our approach can be added to any existing model in a few lines of code, and induces almost no training overhead. We believe SFA provides an excellent tool for practitioners and a strong baseline for research going forward.

Acknowledgements. This work was supported by Vision Semantics Limited, the Alan Turing Institute Turing Fellowship, and the China Scholarship Council.

References

[1] Alexander A Alemi, Ian Fischer, Joshua V Dillon, and Kevin Murphy. Deep variational information bottleneck. *arXiv preprint arXiv:1612.00410*, 2016. 3

[2] Yogesh Balaji, Swami Sankaranarayanan, and Rama Chellappa. Metareg: Towards domain generalization using meta-regularization. In *NeurIPS*, 2018. 2, 5, 6

[3] Gilles Blanchard, Gyemin Lee, and Clayton Scott. Generalizing from several related classification tasks to a new unlabeled sample. In *NIPS*, 2011. 1

[4] Fabio M. Carlucci, Antonio D'Innocente, Silvia Bucci, Barbara Caputo, and Tatiana Tommasi. Domain generalization by solving jigsaw puzzles. In *CVPR*, 2019. 2, 5, 6, 7

[5] Gabriela Csurka. *Domain Adaptation in Computer Vision Applications*. Springer, 2017. 1

[6] Qi Dou, Daniel C. Castro, Konstantinos Kamnitsas, and Ben Glocker. Domain generalization via model-agnostic learning of semantic features. In *NeurIPS*, 2019. 2, 5, 7

[7] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *IJCV*, 2010. 5

[8] Chen Fang, Ye Xu, and Daniel N Rockmore. Unbiased metric learning: On the utilization of multiple datasets and web images for softening bias. In *CVPR*, 2013. 5

[9] Li Fei-Fei, Rob Fergus, and Pietro Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. In *CVPR-W*. IEEE, 2004. 5

[10] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *ICML*, 2017. 5, 6

[11] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In *ICML*, 2015. 5

[12] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *JMLR*, 2016. 5, 6

[13] Jacob R Gardner, Paul Upchurch, Matt J Kusner, Yixuan Li, Kilian Q Weinberger, Kavita Bala, and John E Hopcroft. Deep manifold traversal: Changing labels with convolutional features. *arXiv preprint arXiv:1511.06421*, 2015. 2

[14] Muhammad Ghifary, W. Bastiaan Kleijn, Mengjie Zhang, and David Balduzzi. Domain generalization for object recognition with multi-task autoencoders. In *ICCV*, 2015. 1, 2, 7

[15] Aditya Khosla, Tinghui Zhou, Tomasz Malisiewicz, Alexei Efros, and Antonio Torralba. Undoing the damage of dataset bias. In *ECCV*, 2012. 1, 2

[16] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 3

[17] Pang Wei Koh, Shiori Sagawa, Henrik Marklund, Sang Michael Xie, Marvin Zhang, Akshay Balsubramani, Weihua Hu, Michihiro Yasunaga, Richard Lanas Phillips, Sara Beery, et al. Wilds: A benchmark of in-the-wild distribution shifts. *arXiv preprint arXiv:2012.07421*, 2020. 1, 3

[18] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *NIPS*, 2012. 2

[19] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 1998. 5

[20] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy Hospedales. Deeper, broader and artier domain generalization. In *ICCV*, 2017. 1, 2, 3, 5, 6, 7

[21] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy Hospedales. Learning to generalize: Meta-learning for domain generalization. In *AAAI*, 2018. 2, 5, 6, 7

[22] Da Li, Jianshu Zhang, Yongxin Yang, Cong Liu, Yi-Zhe Song, and Timothy M. Hospedales. Episodic training for domain generalization. In *ICCV*, 2019. 2, 5, 6, 7

[23] Haoliang Li, Sinno Jialin Pan, Shiqi Wang, and Alex C. Kot. Domain generalization with adversarial feature learning. In *CVPR*, 2018. 2, 5, 6, 7

[24] Ya Li, Xinmei Tian, Mingming Gong, Yajing Liu, Tongliang Liu, Kun Zhang, and Dacheng Tao. Deep domain generalization via conditional invariant adversarial networks. In *ECCV*, 2018. 7

[25] Xuanqing Liu, Minhao Cheng, Huan Zhang, and Cho-Jui Hsieh. Towards robust neural networks via random self-ensemble. In *ECCV*, 2018. 3

[26] Bhairav Mehta, Manfred Diaz, Florian Golemo, Christopher J Pal, and Liam Paull. Active domain randomization. In *Conference on Robot Learning*. PMLR, 2020. 2

[27] Saeid Motiian, Marco Piccirilli, Donald A. Adjeroh, and Gianfranco Doretto. Unified deep supervised domain adaptation and generalization. In *ICCV*, 2017. 2, 5, 6, 7

[28] Krikamol Muandet, David Balduzzi, and Bernhard Schölkopf. Domain generalization via invariant feature representation. In *ICML*, 2013. 1, 2

[29] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. In *NeurIPS-W*, 2011. 5

[30] Oren Nuriel, Sagie Benaim, and Lior Wolf. Permuted adain: Enhancing the representation of local cues in image classifiers. *arXiv preprint arXiv:2010.05785*, 2020. 5, 6

[31] Aayush Prakash, Shaad Boochoon, Mark Brophy, David Acuna, Eric Cameracci, Gavriel State, Omer Shapira, and Stan Birchfield. Structured domain randomization: Bridging the reality gap by context-aware synthetic data. In *ICRA*, 2019. 2

[32] Bryan C Russell, Antonio Torralba, Kevin P Murphy, and William T Freeman. Labelme: a database and web-based tool for image annotation. *IJCV*, 2008. 5

[33] Shiv Shankar, Vihari Piratla, Soumen Chakrabarti, Siddhartha Chaudhuri, Preethi Jyothi, and Sunita Sarawagi. Generalizing across domains via cross-gradient training. *arXiv preprint arXiv:1804.10745*, 2018. 2, 5, 6

[34] Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *IROS*, 2017. 2

[35] Hung-Yu Tseng, Hsin-Ying Lee, Jia-Bin Huang, and Ming-Hsuan Yang. Cross-domain few-shot classification via learned feature-wise transformation. In *ICLR*, 2020. 3

[36] Paul Upchurch, Jacob Gardner, Geoff Pleiss, Robert Pless, Noah Snavely, Kavita Bala, and Kilian Weinberger. Deep feature interpolation for image content changes. In *CVPR*, 2017. 2

[37] Vikas Verma, Alex Lamb, Christopher Beckham, Amir Najafi, Ioannis Mitliagkas, David Lopez-Paz, and Yoshua Bengio. Manifold mixup: Better representations by interpolating hidden states. In *ICML*, 2019. 2, 3

[38] Riccardo Volpi, Hongseok Namkoong, Ozan Sener, John C Duchi, Vittorio Murino, and Silvio Savarese. Generalizing to unseen domains via adversarial data augmentation. In *NeurIPS*, 2018. 2

[39] Shujun Wang, Lequan Yu, Caizi Li, Chi-Wing Fu, and Pheng-Ann Heng. Learning from extrinsic and intrinsic supervisions for domain generalization. In *ECCV*, 2020. 2, 5, 6, 7, 8

[40] Jianxiong Xiao, James Hays, Krista A Ehinger, Aude Oliva, and Antonio Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *CVPR*, 2010. 5

[41] Tianyuan Yu, Yongxin Yang, Da Li, Timothy Hospedales, and Tao Xiang. Simple and effective stochastic neural networks. In *AAAI*, 2021. 3

[42] Xiangyu Yue, Yang Zhang, Sicheng Zhao, Alberto Sangiovanni-Vincentelli, Kurt Keutzer, and Boqing Gong. Domain randomization and pyramid consistency: Simulation-to-real generalization without accessing target domain data. In *CVPR*, 2019. 2

[43] Sergey Zakharov, Wadim Kehl, and Slobodan Ilic. Deceptionnet: Network-driven domain randomization. In *CVPR*, 2019. 2

[44] Kaiyang Zhou, Yongxin Yang, Timothy Hospedales, and Tao Xiang. Deep domain-adversarial image generation for domain generalisation. In *AAAI*, 2020. 2, 5, 6, 8

[45] Kaiyang Zhou, Yongxin Yang, Timothy Hospedales, and Tao Xiang. Learning to generate novel domains for domain generalization. In *ECCV*, 2020. 2, 5, 6

[46] Kaiyang Zhou, Yongxin Yang, Yu Qiao, and Tao Xiang. Domain generalization with mixstyle. In *ICLR*, 2021. 2, 3