LATENT FEATURE DISENTANGLEMENT FOR VISUAL DOMAIN GENERALIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Despite remarkable success in a variety of computer vision applications, it is well-known that deep learning can fail catastrophically when presented with out-of-distribution data, where there are usually style differences between the training and test images. Toward addressing this challenge, we consider the domain generalization problem, wherein predictors are trained using data drawn from a family of related training (source) domains and then evaluated on a distinct and unseen test domain. Naively training a model on the aggregate set of data (pooled from all source domains) has been shown to perform suboptimally, since the information learned by that model might be domain-specific and generalize imperfectly to test domains. Data augmentation has been shown to be an effective approach to overcome this problem. However, its application has been limited to enforcing invariance to simple transformations like rotation, brightness change, etc. Such perturbations do not necessarily cover plausible real-world variations that preserve the semantics of the input (such as a change in the image style). In this paper, taking the advantage of multiple source domains, we propose a novel approach to express and formalize robustness to these kinds of real-world perturbations of the images. The three key ideas underlying our formulation are (1) leveraging disentangled representations of the images to define different factors of variations, (2) generating perturbed images by changing such factors composing the representations of the images. (3) enforcing the learner (classifier) to be invariant to such change in the images. We use image to image translation models to demonstrate the efficacy of this framework. Based on this, we propose a *domain-invariant regularization* (**DIR**) loss function, that enforces invariant prediction of targets (class labels) across domains which yields improved generalization performance. We demonstrate the effectiveness of our approach on several widely used datasets for the domain generalization problem, on all of which we achieve competitive results with state-of-the-art models.

1 INTRODUCTION

Deep neural networks (DNNs) have advanced the state-of-the- arts for a wide variety of computer vision applications (Ciregan et al., 2012; Guo et al., 2018; Erhan et al., 2014). The trained models typically perform well on the test/validation data which follows similar characteristics/distribution as the training data, but fail catastrophically when presented with out of distribution (OOD) data in new domains (environments) that may present different characteristics (e.g., style, texture) (Krizhevsky et al., 2012; Taori et al., 2020). The captured images in the new domains in general present style discrepancy with respect to the training data, such as illumination, color contrast/saturation, quality, etc. These result in domain gap/shift between the training (source) and test (target) domains, hence, directly applying a model trained on a source dataset to an unseen target dataset typically suffers from a large performance degradation (Long et al., 2016; Ma et al., 2019).

To address such domain gap/shift issues, numerous research have been conducted that are mainly divided into two categories: Domain Generalization (DG) (Muandet et al., 2013; Li et al., 2018b; Balaji et al., 2018) and Domain Adaptation (DA) (Tzeng et al., 2017; Wang & Deng, 2018). Both DG and DA refer to approaches in which, the model is trained on (multiple) labeled source domains so that it is expected to generalize well to target domains. The key difference between DG and DA is that, DA can access/exploit the unlabeled data of the target domain for training/fine-tuning, while

in DG, the learner does not have access to the target domain data, making the problem much more challenging.

Although a variety of DG approaches have been proposed (Muandet et al., 2013; Li et al., 2018b; Balaji et al., 2018), it was recently shown (Gulrajani & Lopez-Paz, 2021) that no existing domain generalization method can significantly outperform empirical risk minimization (ERM) (Vapnik, 1999) over the training domains when ERM is properly tuned and equipped with state-of-the-art architectures (He et al., 2016) and data augmentation techniques (Gulrajani & Lopez-Paz, 2021).

In this paper, we introduce a new approach for domain generalization based on deep generative image models (Ranzato et al., 2011). Our key idea is to leverage the latent space of a generative model utilizing the data from multiple source domains to capture (latent) domain-specific features of data. We propose *domain-invariant regularization* (**DIR**) loss function that enforce the learner (classifier) to be invariant to such features, making learner more robust under new domains.

More precisely, we assume that there is an underlying (latent) feature space that is a product space of two subspaces: i) **content-specific** feature space containing the semantic information of data (subject-related content), ii) **style-specific** feature space containing domain-related (e.g., style) information of data. We assume the content-specific and the style-specific spaces are disentangled, i.e. style changes are content-preserving. For example, when classifying cats against elephants from images, different parts of the animals constitute content, while style could be, for example, background, lighting conditions and camera lens characteristics. We also assume there is an ideal generator that maps features from the underlying latent space to the image space.

Based on the above assumptions, only content being relevant for the downstream (classification) tasks while the style is irrelevant. Hence, we introduce a regularizer that enforces a classifier to be invariant to the underlying style-specific features of the data.

We utilize image to image translation (I2I) models to learn the generator and the latent features of the data through the training process of generative adversarial networks (GANs) (Creswell et al., 2018). We conduct extensive experiments on several widely used datasets and observe a significant improvement over the naive baseline of training a model on the aggregate dataset from all domains. We also compare **DIR** against other state-of-the-art models and show that our method achieves competitive results. Our contributions are as follows:

- We propose a new objective, Domain Invariant Regularization (**DIR**), that enforces invariant prediction through an explicit regularizer and show improved generalization performance.
- We demonstrate how to leverage I2I models to capture style-specific and content-specific features of data, thus allowing us to automatically generate realistic content preserving variations in data.
- We demonstrate how to incorporate the categorical semantic features (such as object labels) into the content-specific feature space using the source domains categorical (class) labels. We learn such features from multiple object categories shared between perceptually different domains by incorporating a category-label classifier into I2I models.
- We show the effectiveness of our method by performing extensive experiments on widely used domain generalization datasets (e.g., PACS (Wang et al., 2020a), Office-Home (Venkateswara et al., 2017), DomainNet (Peng et al., 2019)) and comparing with relevant state-of-the-art baselines.

2 Related Work

Many DG methods (Huang et al., 2021; Xu et al., 2021; Zhao et al., 2020; Mahajan et al., 2021; Muandet et al., 2013; Zhou et al., 2021; Li et al., 2018b; Balaji et al., 2018; Wang et al., 2020a) aim to learn a domain-invariant feature representation or classifier across the source domains, in the hope that it would also be invariant to domain shift brought by the target domain. Other approaches include meta learning (Li et al., 2018a), invariant risk minimization (Arjovsky et al., 2019), distributionally robust optimization (Sagawa et al., 2019), mixup (Wang et al., 2020b), and causal matching (Mahajan et al., 2021). Adversarial training with improvements (Pei et al., 2018) has also been used to learn invariant representations. Li et al. (2018a) propose a meta-learning solution, which uses a model agnostic training procedure to simulate train/test domain shift during training and jointly optimize

the simulated training and test domains within each mini-batch. However, there is an intrinsic flaw in this approach, that is, when the source domains become more diverse, learning a domain invariant model becomes more difficult. This is because each domain now contains much domain-specific information.

Complementary to these approaches, we focus instead on learning invariant classifiers[1] by explicitly enforcing soft invariance-based constraints on the classifiers.

Another common pervasive theme in domain generalization literature is to use DA techniques to transform the source data into a lower-dimensional feature space that is invariant to domains but retains the discriminative class information. Domain-invariant features could be learned by directly minimizing distribution divergence measures, such as MMD (Long et al., 2015; 2017) or optimal transport (Damodaran et al., 2018; Balaji et al., 2019).

The multi-source DA methods are more related to this work because of the same problem setting. Xu et al. (2018); Peng et al. (2019); Li et al. (2018b) extended the alignment idea to multi-source DA by considering all possible source-target distance pairs. Relationships between each source and the target were exploited in (Li et al., 2018c) where only the target-related sources were kept for model learning. Hoffman et al. (2018) computed distribution-based weights for combining source classifiers.

Another approach toward improving OOD performance is to modify or augment the available training data. Data augmentation (Hoffer et al., 2020; Liu et al., 2020; Ko & Gu, 2020; Lee et al., 2020; Lin et al., 2019) is a well-known approach to tackle this issue, encoding additional priors in the form of invariant feature transformations. Intuitively, data augmentation enables the model to train on more data, encouraging the model to capture certain types of invariance with respect to its inputs and outputs leading to better generalization performance; data augmentation may also produce data that may be closer to an out-of-distribution target task. For example, Mixup also uses information from two images. Rather than implanting one portion of an image inside another, Mixup produces an element-wise convex combination of two images (Zhang et al., 2017). The AugMix (Hendrycks et al., 2020) aims to make models robust to out-of-distribution data by exposing the model to a wide variety of augmented images. In Augmix, several augmentation 'chains' are sampled, where a chain is a composition of one to three randomly selected augmentation operations (e.g., rotation, scale, contrast, etc).

3 NOTATION

Throughout our paper, we use $\mathbf{x} \in \mathcal{X}, \mathbf{y} \in \mathcal{Y}$, and $\mathbf{d} \in \mathcal{D}$ to denote the input image, its class label, and its domain label taking values from the image space $\mathcal{X}$, the class label space $\mathcal{Y}$, and the domain label space $\mathcal{D}$ respectively. Each training data point $(\mathbf{x}, \mathbf{y}, \mathbf{d})$ is sampled from an unknown joint distribution $\mathbb{P}(\mathbf{x}, \mathbf{y}, \mathbf{d})$ over $\mathcal{X} \times \mathcal{Y} \times \mathcal{D}$. y also denotes the y-th coordinate of the on-hot class label $\mathbf{y}$

$\mathcal{Z}$ denotes the underlying latent space that is a product space of the form $\mathcal{Z} = \mathcal{Z}_c \times \mathcal{Z}_s$, where $\mathcal{Z}_c$ denotes the **content-specific** feature space and $\mathcal{Z}_s$ denotes the **style-specific** feature space. The generator $\mathcal{G} : \mathcal{Z} \to \mathcal{X}$ maps features form $\mathcal{Z}$ to the data (image) space $\mathcal{X}$. We also denote the classifier by $f : \mathcal{X} \to \mathcal{Y}$ that maps the samples from the image space $\mathcal{X}$ to their class label in the output space $\mathcal{Y}$.

4 METHODOLOGY

4.1 PROBLEM DEFINITION

For the Domain Generalization (DG) task, we assume we have access to i.i.d. data from S source domains $\{\mathbf{D}_1, \cdots, \mathbf{D}_S\}$. The goal of DG is to learn a classifier f that generalizes well to unseen target domain(s) where no data about the target domain is available during training (out-of-domain generalization), as well as new data from existing domains (in-domain generalization) (Shankar et al., 2018).

[1]Note that our approach is not opposed to invariant feature learning, we merely focus on learning domain invariant classifiers, that can be easily combined with domain-invariant feature learning approaches.

4.2 PROPOSED SOLUTION

In general, learning a robust predictive model f that is invariant across different domains with different data distributions has always been challenging. We must assume the existence of some statistical invariances across training and test domains in order to incorporate such invariances into the classifier. Assuming a disentangled latent space (Higgins et al., 2018), we hypothesize that there exists a latent subspace that is domain invariant. We propose a generative model with two independent sources of variation; $\mathbf{z}_s \in \mathcal{Z}_s$, which is style-specific containing domain-related (e.g., style, background) information of data, and $\mathbf{z}_c \in \mathcal{Z}_c$ which is content-specific containing the semantic information of data (subject-related content). A generator $\mathcal{G} : \mathcal{Z} \to \mathcal{X}$ maps features form $\mathcal{Z} = \mathcal{Z}_c \times \mathcal{Z}_s$ to the data (image) space $\mathcal{X}$. For a given classification task where the goal is to predict the label $\mathbf{y}$ for a data point $\mathbf{x}$, only the feature $\mathbf{z}_c$ is relevant, while $\mathbf{z}_s$ is irrelevant. The **causal** graphical model representing the causal relationship between a sample $(\mathbf{x}, \mathbf{y}, \mathbf{d})$, and its latent features is illustrated in Fig. 1.

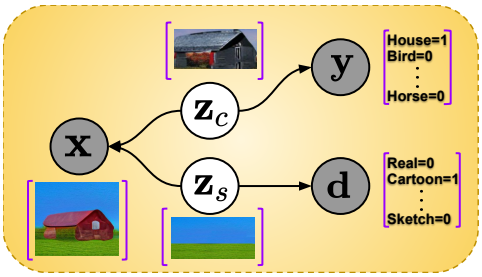

Figure 1: Causal graph illustrating assumptions about content-specific feature $\mathbf{z}_c$ and style-specific feature $\mathbf{z}_s$ of the data point $\mathbf{x}$ and their relationship with category label $\mathbf{y}$ and domain label $\mathbf{d}$. Observed variables are shaded.

In Fig. 1, the directed arrows from $\mathbf{z}_c$ and $\mathbf{z}_s$ to the observed data $\mathbf{x}$ (e.g. images) indicate that $\mathbf{x}$ is generated based on content and style. The directed arrow from $\mathbf{z}_c$ to the class label $\mathbf{y}$ encodes the assumption that content $\mathbf{z}_c$ (e.g., shape) directly influences the class label, while the absence of any directed arrow from $\mathbf{z}_s$ to $\mathbf{y}$ indicates that style does not (it should be noted that style features may be correlated to class labels, but not causally related to them). Thus, content $\mathbf{z}_c$ has all the necessary information to predict $\mathbf{y}$. Similarly, the directed arrow from $\mathbf{z}_s$ to the domain label $\mathbf{d}$ implies the style $\mathbf{z}_s$ directly influences the domain label, while the content $\mathbf{z}_c$ does not. Thus, the style $\mathbf{z}_s$ encodes all the information of the domain (e.g., appearance, background). The absence of any directed path between $\mathbf{z}_c$ and $\mathbf{z}_s$ in Fig. 1 encodes the intuition that these variables are marginally independent, i.e. $\mathbf{z}_c \perp\!\!\!\perp \mathbf{z}_s$.

Based on our causal relationship assumption shown in Fig. 1, it is clear that robustifying f against spurious features $\mathbf{z}_s$ associated with class labels, is a seemingly plausible way to improve the classifier generalization ability to unseen domains.

We formalize it using conditional independence: given a sample $\mathbf{x} = \mathcal{G}(\mathbf{z}_c, \mathbf{z}_s)$ with $\mathbf{z}_c \in \mathcal{Z}_c$ and $\mathbf{z}_s \in \mathcal{Z}_s$, and its corresponding label $\mathbf{y}$, we have

$$\mathbb{P}(\mathbf{y}|\mathbf{z_c}, \mathbf{z_s}) = \mathbb{P}(\mathbf{y}|\mathbf{z_c}). \tag{1}$$

In other words, given an input image $\mathbf{x}$, manipulating its style feature $\mathbf{z_s}$ does not influence its class label. Hence, the ideal invariant classifier f^* that outputs a probability distribution over $\mathcal{Y}$ should be consistent with the invariance assumption

$$f^*(\mathcal{G}(\mathbf{z}_c, \mathbf{z}_s)) = f^*(\mathcal{G}(\mathbf{z}_c, \tilde{\mathbf{z}}_s)), \quad \forall\, \tilde{\mathbf{z}}_s \in \mathcal{Z}_s. \tag{2}$$

To achieve invariant prediction, we propose to explicitly enforce invariance under style perturbations through a regularizer we call Domain Invariant Regularization (DIR) loss. We write this as

$$\mathcal{L}_{reg} = \mathbb{E}_{\mathbf{x}\sim\mathbb{P}(\mathbf{x})}\mathbb{E}_{\mathbf{d}\sim\mathbb{P}(\mathbf{d})}\mathbb{E}_{\mathbf{u}\sim\mathcal{N}(0,I)}\big[\mathbb{D}(f(\mathbf{x}), f(\tilde{\mathbf{x}}))\big], \quad \tilde{\mathbf{x}} = \mathcal{G}\big(\mathcal{F}(\mathbf{x}), \mathcal{T}(\mathbf{d}, \mathbf{u})\big), \tag{3}$$

where $\mathcal{F} : \mathcal{X} \to \mathcal{Z}_c$ is a function that maps data points to their content-specific features, and $\mathcal{T} : \mathcal{D} \times \mathbb{R}^n \to \mathcal{Z}_s$ is a function that takes a domain index $\mathbf{d}$ and map a n-dimensional Gaussian vector $\mathbf{u} \in \mathbb{R}^n$ to a point on $\mathcal{Z}_s$ of that domain (we discuss in Sec. 4.3 how to learn these functions). $\mathbb{D}(p_1, p_2)$ is a distance measure between two probability vectors p_1 and p_2. In this study, we utilize the L_1 distance (the absolute values of the difference between the classifier's probabilistic outputs) as:

$$\mathbb{D}(p_1, p_2) = \sum_{i=1}^{K} |p_1^k - p_2^k|, \tag{4}$$

where p_1^k and p_2^k denote probability output of p_1 and p_2 for class k respectively. Note that any distance measure on distributions can be used in place of the L_1 distance. Intuitively, **DIR** encourages the

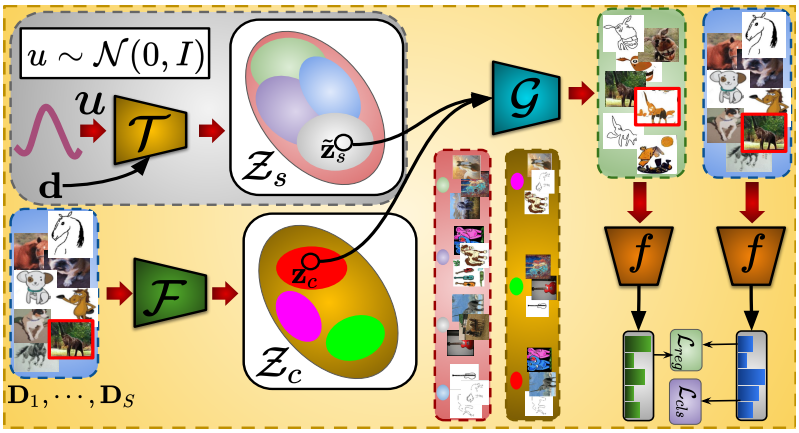

Figure 2: An overview of the proposed model. For a source sample $(\mathbf{x}, \mathbf{y})$ (red box), its content-specific feature $\mathbf{z}_c$ is extracted through the mapping $\mathcal{F}$. By randomly picking a domain index $\mathbf{d}$, and drawing a Gaussian vector, a random domain feature $\tilde{\mathbf{z}}_s$ is generated by $\mathcal{T}$. A new (semantically preserved) sample $\tilde{\mathbf{x}}$ is then generated by feeding $\mathbf{z}_c$ and $\tilde{\mathbf{z}}_s$ into $\mathcal{G}$. The classifier f is trained by two losses (all $\mathcal{T}$, $\mathcal{F}$, and $\mathcal{G}$ are pretrained and fixed during training f): 1) The standard cross-entropy loss $\mathcal{L}_{cls}$ that encourages f to predict the correct class label for $\mathbf{x}$, 2) The domain-invariant regularization loss $\mathcal{L}_{reg}$ that encourages f to make similar prediction for $\tilde{\mathbf{x}}$ and $\mathbf{x}$.

classifier f to be invariant to the induced semantically irrelevant perturbations to the data that arise from altering the input samples through plausible style perturbations. These perturbations to the input are meaningful by using a disentangled latent feature that encodes independent controllable factors, where style-specific factors are known to be independent from the class label. The final objective can be written as

$$f^* = \arg\min_f \mathcal{L}_{cls} + \lambda \mathcal{L}_{reg}, \tag{5}$$

where λ is a hyper-parameter to control a trade-off between the classifier's prediction accuracy on the source samples and the classifier's consistency over the samples' perturbations. $\mathcal{L}_{cls}$ also denotes the standard multi-class cross-entropy loss function defined as

$$\mathcal{L}_{cls} = \mathbb{E}_{(\mathbf{x},\mathbf{y})\sim\mathbb{P}(\mathbf{x},\mathbf{y})}\big[-\log([f(\mathbf{x})]_y)\big], \tag{6}$$

where $[a]_i$ returns the i-th coordinate of a. The **DIR** approach is depicted in Fig. 2.

4.3 Training Methodology

We learn $\mathcal{T}, \mathcal{F}, \mathcal{G}$, and f in two separate steps. First, we learn $\mathcal{T}, \mathcal{F}, \mathcal{G}$, then, we fix them, and learn f.

4.3.1 Learning $\mathcal{T}, \mathcal{F}$, and $\mathcal{G}$

So far, we have assumed the presence of a domain-specific generator $\mathcal{T}$, a content specific encoder $\mathcal{F}$ and an image generator $\mathcal{G}$. This section details how to train such functions using I2I models.

To do so, we train multi-domain image-to-image translation networks (MI2I) on the instances drawn from the training domains. MI2I models are designed to transform images between distinct datasets so that they resemble a diverse collection of images from another dataset.

The architectures of MI2I models generally consist of two components: a disentangled feature extraction module and a image generation module. The role of feature extraction module is to recover the style-specific and content-specific features of the images and the goal of image generation module given a sample image $\mathbf{x}$ and a target domain $\mathbf{d}$ is to generate an instance in domain $\mathbf{d}$ using the extracted content feature of $\mathbf{x}$ and a randomly generated style feature in $\mathbf{d}$. In this way, MI2I are a natural framework for learning $\mathcal{T}$, $\mathcal{F}$, and $\mathcal{G}$. In particular, we use the StarGAN loss functions (Saito et al., 2019) to learn $\mathcal{T}$, $\mathcal{F}$, and $\mathcal{G}$ (the details are available in Appendix).

Although StarGAN model learns accurate and diverse transformations between multiple source domain, it can consequently result in arbitrary mappings as the translation is done without supervision

between domains that share common semantic attributes (e.g. class labels). In other works, MI2I models commonly been applied on domains in which a translation entails little geometric changes and the style of the generated image is independent of the semantic content in the source sample (e.g., translating horses $\rightarrow$ zebras). In order to leverage the category labels of source samples, we propose **C-StarGAN** by incorporating a classification module $\mathcal{C} : \mathcal{X} \rightarrow \mathcal{Y}$ into the StarGAN model. During the training of the C-StarGAN, the classifier is only trained on the actual labeled source samples using the cross entropy loss. The $\mathcal{G}$ and $\mathcal{F}$ are then trained to translate input images to new domains belonging to their own classes by encouraging them to minimize the cross entropy loss on the generated images (see the Appendix for more details).

4.3.2 LEARNING f

Given N training samples $\{\mathbf{x}_i, \mathbf{y}_i, \mathbf{d}_i\}_{i=1}^{N}$ from S source domains, we approximate the expectations in Eqs. 3 and 6 with empirical average and obtain f as

$$f^* = \arg\min_f \frac{1}{N} \sum_{i=1}^{N} \left[-\log([f(\mathbf{x}_i)]_{y_i}) + \lambda \sum_{j=1}^{M} \mathbb{D}\Big(f(\mathbf{x}_i) - f(\tilde{\mathbf{x}}_j) \Big) \right], \tag{7}$$

where

$$\tilde{\mathbf{x}}_i = \mathcal{G}\big(\mathcal{F}(\mathbf{x}_i), \mathcal{T}(\tilde{\mathbf{d}}_j, \tilde{\mathbf{u}}_j)\big), \;\; \tilde{\mathbf{d}}_j \sim \mathrm{U}(\{1, 2, ..., S\}), \;\; \tilde{\mathbf{u}}_j \sim \mathcal{N}(0, I), \tag{8}$$

and $U(.)$ denote discrete uniform distribution over the index set $\{1, 2, ..., S\}$. Intuitively, for each training image $\mathbf{x}_i$, we encourage f (i) to correctly predict its class label $\mathbf{y}_i$, (ii) to have similar prediction with a set of M perturbed images $\{\tilde{\mathbf{x}}_j\}_{j=1}^{M}$ with the same content as $\mathbf{x}_i$ under varying style. The training procedures is detailed in Algorithm 1 in the Appendix.

5 EXPERIMENTAL RESULTS

5.1 DATASETS

To evaluate our method, we perform experiments on three datasets that are commonly used in the literature for domain generalization (some sample images from each domain is shown in Fig. 3).

PACS (Wang et al., 2020a) contains $9,991$ images from four different domains: Art-Painting, Cartoon, Photo, Sketch. The task is classification with seven classes.

OfficeHome (Venkateswara et al., 2017) has $15,500$ images of daily objects from four domains: art, clipart, product and real. There are 65 classes in this classification dataset.

DomainNet (Peng et al., 2019) is a recently introduced benchmark for large-scale multi-source domain adaptation. It has six domains (Clipart, Infograph, Painting, Quickdraw, Real and Sketch) and 0.6M images of 345 classes. The full DomainNet requires considerable computing resources for training. Following (Zhou et al., 2021), we use **mini-DomainNet**, which takes a subset of DomainNet (four domains and 126 classes) containing $18,703$ images of Clipart, $31,202$ images of Painting, $65,609$ images of Real and $24,492$ images of Sketch.

5.2 BASELINES

We compare **DIR** with various recent algorithms on domain generalization problem. we used the DomainBed (Gulrajani & Lopez-Paz, 2021) package, facilitating comparison to a range of state-of-the-art methods namely **ERM** (Vapnik, 1999),**DRO** (Sagawa et al., 2019), **Mixup** (Zhang et al., 2017), **MLDG** (Li et al., 2018a), **CORAL** (Peng et al., 2019), **MMD** (Lee et al., 2019), **Sagnet** (Nam et al., 2019), and **MTL** (Blanchard et al., 2021).

5.3 EXPERIMENTAL SETTING

For all datasets, we perform "leave-one-domain-out" experiments, where we choose one domain as the target domain, train the model on all remaining domains and evaluate it on the chosen domain. For

(a) PACS datasets (b) OfficeHome dataset (c) Mini-DomainNet dataset

Figure 3: Exemplary images from different datasets. a) **PACS** datatset (first row: Art-painting, second row: Cartoon, Third row: Photo, last row: Sketch), b) **OfficeHome** dataset (first column: Art, second column: ClipArt, Third column: Product, last column: Photo). c) **Mini-DomainNet** dataset (first column: ClipArt, second column: Painting, Third column: Real, last column: Sketch).

Table 1: Leave-one-domain-out generalization results on PACS dataset. The best (green), the second best (blue).

Algorithm	**Art-Painting**	**Cartoon**	**Photo**	**Sketch**	**Avg**
ERM	81.3 ± 0.5	80.1 ± 0.5	95.4 ± 0.4	79.4 ± 0.2	84.05
DRO	81.7 ± 0.4	80.0 ± 0.3	94.5 ± 0.3	79.6 ± 0.2	83.94
Mixup	81.9 ± 0.5	79.4 ± 0.1	95.6 ± 0.8	78.7 ± 0.1	84.23
MLDG	81.6 ± 0.6	80.0 ± 0.1	94.8 ± 0.8	80.2 ± 0.4	84.10
CORAL	81.1 ± 0.2	80.5 ± 0.4	95.2 ± 0.8	79.2 ± 0.2	84.27
MMD	80.6 ± 0.2	80.7 ± 0.1	94.9 ± 1.4	79.1 ± 0.8	83.87
MTL	80.4 ± 0.8	78.0 ± 0.5	94.0 ± 0.8	77.0 ± 1.5	82.35
SagNet	80.0 ± 1.0	80.4 ± 0.6	94.6 ± 0.1	79.5 ± 0.4	83.62
DIR (Ours)	85.0 ± 1.5	81.4 ± 0.5	95.8 ± 0.8	82.3 ± 0.1	86.10

all datasets, we use a Resnet18 (He et al., 2016) network as the classifier f. As a standard practice, the Resnet18 backbone is pre-trained on ImageNet.

Data augmentation is also standard practice for real-world computer vision datasets, and during the training we augment our data as follows: crops of random size and aspect ratio, resizing to 224×224 pixels, random horizontal flips, random color jitter, randomly converting the image tile to grayscale with 10% probability, and normalization using the ImageNet channel means and standard deviations.

Hyperparameter Search: Following standard practice, we use 90% of available data as training data and 10% as validation data to select hyper-parameter λ.
Random Trials: For fair comparison, for each target domain, we have reported our average and standard deviation for five independent runs of the model.
The StarGAN (Choi et al., 2020) model implementation is taken from the authors' original source code[2]. Further details concerning hyperparameter tuning and model selection (e.g. batch-size, learning rate, etc.) are deferred to the Appendix.

5.4 Results

Results are summarized in Tables 1,2, and 3 where each experiment is averaged over 5 independent trials. We achieve SOTA results on all three datasets that verifies the effectiveness of the proposed approach. As can be seen **DIR** provides an average improvement of 3.2% in accuracy across all three datasets over the **ERM** baseline that simply combines data from all the training domains to train a model. This can be attributed to the increased diversity in the training data in each domain by perturbing samples by modifying their style, which allows us to construct images that cover the space of domains better. Moreover, according to the assumption of the existence of disentangled latent space, when the number of training domains is appropriate, **DIR** can mitigate the impact of spurious correlation between the domain-specific features and the class labels by relying only on the content-specific features to achieve more effective domain generalization. In comparison to other DG

[2]https://github.com/clovaai/stargan-v2

Table 2: Leave-one-domain-out generalization results on OfficeHome dataset. The best (green), the second best (blue).

Algorithm	Art	ClipArt	Product	Photo	Avg
ERM	52.0 ± 0.6	49.0 ± 0.3	69.0 ± 0.2	70.1 ± 0.2	60.13
DRO	51.2 ± 0.6	47.4 ± 0.3	68.7 ± 0.2	69.3 ± 0.2	58.78
Mixup	53.0 ± 0.9	51.2 ± 0.3	71.5 ± 1.6	72.2 ± 0.3	62.03
MLDG	52.5 ± 0.7	49.0 ± 0.3	69.4 ± 1.2	70.1 ± 0.1	60.45
CORAL	55.8 ± 0.6	52.0 ± 0.2	71.5 ± 1.3	74.5 ± 0.2	63.11
MMD	52.6 ± 0.3	49.9 ± 0.5	69.8 ± 0.5	70.6 ± 0.9	60.83
MTL	51.5 ± 0.3	50.1 ± 0.8	70.4 ± 0.4	73.3 ± 0.4	61.32
SagNet	53.4 ± 0.7	52.3 ± 0.3	70.6 ± 0.3	74.1 ± 0.4	62.60
DIR (Ours)	56.2 ± 1.4	55.1 ± 0.4	72.4 ± 1.8	74.0 ± 0.2	64.45

approaches, **DIR** either performs better or competitively with the best approach on each individual dataset.

In order to evaluate the proposed approach for in-domain generalization setting (the test set contains a mix of unseen samples from source domains), we conducted an experiment, in which we split the source domain samples into training and test sets with various proportions. Then, we trained a model on source training samples and tested it on source test samples. Due to lack of space, the results are available in the Appendix.

5.5 Ablation Studies

5.5.1 Sensitivity Analysis to Parameter λ

To analyze the sensitivity of **DIR** to changes in parameter λ, we conducted additional experiments to analyze the parameter sensitivity of **DIR** w.r.t. the various values of λ. To this end, we consider OfficeHome dataset here. Fig. 4 shows the sensitivity analysis of **DIR** respect to λ. Sensitivity analysis is performed by varying λ at the time over a given range, while for the other parameters we set them to their final values. From Fig. 4, we see that when $\lambda = 0$ (no domain-invariant regularization term is considered), we recover the **ERM** method for which the performance drops considerably. For other values of λ, the performance is superior and there is little variation in the model performance, evidencing the robustness of our method w.r.t. λ.

5.5.2 Effectiveness of C-StarGAN

To verify the effectiveness of the proposed C-StarGAN model, we compare both qualitatively and quantitatively with StarGAN. The experiment is conducted on PACS and the average performance

Table 3: Leave-one-domain-out generalization results on mini-DomainNet datasets. The best (green), the second best (blue).

Algorithm	ClipArt	Painting	Real	Sketch	Avg
ERM	65.5 ± 0.3	57.1 ± 0.5	62.3 ± 0.2	57.1 ± 0.1	60.50
DRO	64.8 ± 0.4	57.4 ± 0.4	61.5 ± 0.5	56.9 ± 0.1	60.15
Mixup	67.1 ± 0.2	59.1 ± 0.5	64.3 ± 0.3	59.2 ± 0.3	62.42
MLDG	65.7 ± 0.2	57.0 ± 0.2	63.7 ± 0.3	58.1 ± 0.1	61.12
CORAL	66.5 ± 0.2	59.5 ± 0.4	66.0 ± 0.6	59.5 ± 0.1	62.87
MMD	65.0 ± 0.5	58.0 ± 0.2	63.8 ± 0.2	58.4 ± 0.7	61.30
MTL	65.3 ± 0.5	59.0 ± 0.4	65.6 ± 0.4	58.5 ± 0.2	62.10
SagNet	65.0 ± 0.4	58.1 ± 0.2	64.2 ± 0.3	58.1 ± 0.4	61.35
DIR (Ours)	68.2 ± 0.3	60.5 ± 0.3	65.8 ± .4	60.0 ± 0.1	63.62

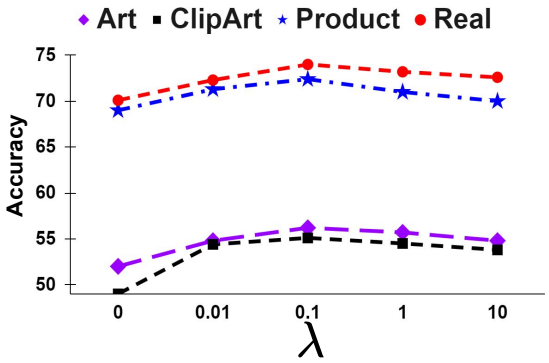

Figure 4: Sensitivity analysis of **DIR** to the hyper-parameter λ on OfficeHome dataset.

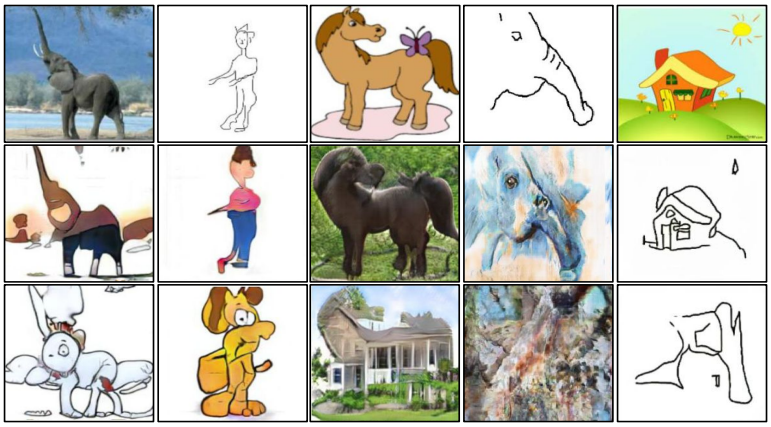

Figure 5: Image generation results on PACS. First row: Original Images. Second row: Transformed Images using C-StarGAN. Last row: Transformed Images using StarGAN.

over test domains is used for comparison. Tab. 4 shows that training $\mathcal{F}, \mathcal{T}, \mathcal{G}$ using StarGAN performs slightly better than the **ERM** model (StarGAN's 84.55% vs. **ERM**'s 84.05%) while training them using C-StarGAN obtains a clear improvement of 1.55% over StarGAN. This confirms C-StarGAN learns a better domain and content disentanglement than StarGAN. Fig. 5 shows some examples of input images and their corresponding generated images in other domains. As shown in Fig. 5, without the classifier, the StarGAN does not preserve the semantic content of the input images, while C-StarGAN successfully capture such information to translate input images to new domains.

Model	Art-Painting	Cartoon	Photo	Sketch	Avg
DIR (StarGAN)	82.7	80.2	95.0	80.4	84.55
DIR(C-StarGAN)	**85.0**	**81.4**	**95.8**	**82.3**	**86.10**

Table 4: Domain Generalization results for DIR using StarGAN and C-StarGAN on PACS dataset.

6 Conclusion

In this paper, we introduced a new approach for domain generalization by proposing a new regularizer called Domain Invariant Regularization (**DIR**). In this approach, we showed that under latent disentanglement assumption, we can diminish the effect of spurious features on training the classifier, encouraging it to rely on discriminative domain-invariant features. We then introduced an implementation for our approach in practice with the domain transformations learned by the StarGAN model and empirically showed that our approach outperforms other state-of-the-art models on several datasets. In the future, We plan to extend the domain-invariant learning framework to the more challenging applications such as visual Semantic Segmentation and Object Detection.

REFERENCES

Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. Invariant risk minimization. *arXiv preprint arXiv:1907.02893*, 2019.

Yogesh Balaji, Swami Sankaranarayanan, and Rama Chellappa. Metareg: Towards domain generalization using meta-regularization. *Advances in Neural Information Processing Systems*, 31: 998–1008, 2018.

Yogesh Balaji, Rama Chellappa, and Soheil Feizi. Normalized wasserstein for mixture distributions with applications in adversarial learning and domain adaptation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 6500–6508, 2019.

Gilles Blanchard, Aniket Anand Deshmukh, Ürün Dogan, Gyemin Lee, and Clayton Scott. Domain generalization by marginal transfer learning. *J. Mach. Learn. Res.*, 22:2–1, 2021.

Yunjey Choi, Youngjung Uh, Jaejun Yoo, and Jung-Woo Ha. Stargan v2: Diverse image synthesis for multiple domains. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8188–8197, 2020.

Dan Ciregan, Ueli Meier, and Jürgen Schmidhuber. Multi-column deep neural networks for image classification. In *2012 IEEE conference on computer vision and pattern recognition*, pp. 3642–3649. IEEE, 2012.

Antonia Creswell, Tom White, Vincent Dumoulin, Kai Arulkumaran, Biswa Sengupta, and Anil A Bharath. Generative adversarial networks: An overview. *IEEE Signal Processing Magazine*, 35(1): 53–65, 2018.

Bharath Bhushan Damodaran, Benjamin Kellenberger, Rémi Flamary, Devis Tuia, and Nicolas Courty. Deepjdot: Deep joint distribution optimal transport for unsupervised domain adaptation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 447–463, 2018.

Dumitru Erhan, Christian Szegedy, Alexander Toshev, and Dragomir Anguelov. Scalable object detection using deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2147–2154, 2014.

Ishaan Gulrajani and David Lopez-Paz. In search of lost domain generalization. *ICLR*, 2021.

Yanming Guo, Yu Liu, Theodoros Georgiou, and Michael S Lew. A review of semantic segmentation using deep neural networks. *International journal of multimedia information retrieval*, 7(2):87–93, 2018.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Dan Hendrycks, Norman Mu, Ekin D Cubuk, Barret Zoph, Justin Gilmer, and Balaji Lakshminarayanan. Augmix: A simple data processing method to improve robustness and uncertainty. *International Conference on Learning Representation (ICLR)*, 2020.

Irina Higgins, David Amos, David Pfau, Sebastien Racaniere, Loic Matthey, Danilo Rezende, and Alexander Lerchner. Towards a definition of disentangled representations. *arXiv preprint arXiv:1812.02230*, 2018.

Elad Hoffer, Tal Ben-Nun, Itay Hubara, Niv Giladi, Torsten Hoefler, and Daniel Soudry. Augment your batch: Improving generalization through instance repetition. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

Judy Hoffman, Mehryar Mohri, and Ningshan Zhang. Algorithms and theory for multiple-source adaptation. *arXiv preprint arXiv:1805.08727*, 2018.

Jiaxing Huang, Dayan Guan, Aoran Xiao, and Shijian Lu. Fsdr: Frequency space domain randomization for domain generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6891–6902, 2021.

Byungsoo Ko and Geonmo Gu. Embedding expansion: Augmentation in embedding space for deep metric learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25:1097–1105, 2012.

Chen-Yu Lee, Tanmay Batra, Mohammad Haris Baig, and Daniel Ulbricht. Sliced wasserstein discrepancy for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10285–10295, 2019.

Donghoon Lee, Hyunsin Park, Trung Pham, and Chang D. Yoo. Learning augmentation network via influence functions. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M Hospedales. Learning to generalize: Meta-learning for domain generalization. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018a.

Haoliang Li, Sinno Jialin Pan, Shiqi Wang, and Alex C Kot. Domain generalization with adversarial feature learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5400–5409, 2018b.

Yitong Li, Michael Murias, Samantha Major, Geraldine Dawson, and David E Carlson. Extracting relationships by multi-domain matching. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pp. 6799–6810, 2018c.

Chen Lin, Minghao Guo, Chuming Li, Xin Yuan, Wei Wu, Junjie Yan, Dahua Lin, and Wanli Ouyang. Online hyper-parameter learning for auto-augmentation strategy. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

Qingfeng Liu, Behnam Gholami, Mostafa El-Khamy, and Jungwon Lee. Diversification is all you need: Towards data efficient image understanding. *Technical Report*, 2020.

Mingsheng Long, Yue Cao, Jianmin Wang, and Michael Jordan. Learning transferable features with deep adaptation networks. In *International conference on machine learning*, pp. 97–105. PMLR, 2015.

Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I Jordan. Unsupervised domain adaptation with residual transfer networks. *NIPS*, 2016.

Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I Jordan. Deep transfer learning with joint adaptation networks. In *International conference on machine learning*, pp. 2208–2217. PMLR, 2017.

Xinhong Ma, Tianzhu Zhang, and Changsheng Xu. Deep multi-modality adversarial networks for unsupervised domain adaptation. *IEEE Transactions on Multimedia*, 21(9):2419–2431, 2019.

Divyat Mahajan, Shruti Tople, and Amit Sharma. Domain generalization using causal matching. In *International Conference on Machine Learning*, pp. 7313–7324. PMLR, 2021.

Krikamol Muandet, David Balduzzi, and Bernhard Schölkopf. Domain generalization via invariant feature representation. In *International Conference on Machine Learning*, pp. 10–18. PMLR, 2013.

Hyeonseob Nam, HyunJae Lee, Jongchan Park, Wonjun Yoon, and Donggeun Yoo. Reducing domain gap via style-agnostic networks. *arXiv preprint arXiv:1910.11645*, 2(7):8, 2019.

Zhongyi Pei, Zhangjie Cao, Mingsheng Long, and Jianmin Wang. Multi-adversarial domain adaptation. In *Thirty-second AAAI conference on artificial intelligence*, 2018.

Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1406–1415, 2019.

Marc'Aurelio Ranzato, Joshua Susskind, Volodymyr Mnih, and Geoffrey Hinton. On deep generative models with applications to recognition. In *CVPR 2011*, pp. 2857–2864. IEEE, 2011.

Shiori Sagawa, Pang Wei Koh, Tatsunori B Hashimoto, and Percy Liang. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. *arXiv preprint arXiv:1911.08731*, 2019.

Kuniaki Saito, Donghyun Kim, Stan Sclaroff, Trevor Darrell, and Kate Saenko. Semi-supervised domain adaptation via minimax entropy. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 8050–8058, 2019.

Shiv Shankar, Vihari Piratla, Soumen Chakrabarti, Siddhartha Chaudhuri, Preethi Jyothi, and Sunita Sarawagi. Generalizing across domains via cross-gradient training. *ICML*, 2018.

Rohan Taori, Achal Dave, Vaishaal Shankar, Nicholas Carlini, Benjamin Recht, and Ludwig Schmidt. Measuring robustness to natural distribution shifts in image classification. *NIPS*, 2020.

Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7167–7176, 2017.

Vladimir N Vapnik. An overview of statistical learning theory. *IEEE transactions on neural networks*, 10(5):988–999, 1999.

Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5018–5027, 2017.

Mei Wang and Weihong Deng. Deep visual domain adaptation: A survey. *Neurocomputing*, 312: 135–153, 2018.

Shujun Wang, Lequan Yu, Caizi Li, Chi-Wing Fu, and Pheng-Ann Heng. Learning from extrinsic and intrinsic supervisions for domain generalization. In *European Conference on Computer Vision*, pp. 159–176. Springer, 2020a.

Yufei Wang, Haoliang Li, and Alex C Kot. Heterogeneous domain generalization via domain mixup. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3622–3626. IEEE, 2020b.

Qinwei Xu, Ruipeng Zhang, Ya Zhang, Yanfeng Wang, and Qi Tian. A fourier-based framework for domain generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14383–14392, 2021.

Ruijia Xu, Ziliang Chen, Wangmeng Zuo, Junjie Yan, and Liang Lin. Deep cocktail network: Multi-source unsupervised domain adaptation with category shift. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3964–3973, 2018.

Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *International Conference on Learning Representation (ICLR)*, 2017.

Shanshan Zhao, Mingming Gong, Tongliang Liu, Huan Fu, and Dacheng Tao. Domain generalization via entropy regularization. *Advances in Neural Information Processing Systems*, 33, 2020.

Kaiyang Zhou, Yongxin Yang, Yu Qiao, and Tao Xiang. Domain adaptive ensemble learning. *IEEE Transactions on Image Processing*, 30:8008–8018, 2021.

A APPENDIX

This Appendix consists of the following parts:

- Appendix B: Quantitative results investigating the in-domain generalization performance of **DIR** and comparing it with other baselines.
- Appendix C: The details of learning $\mathcal{F}, \mathcal{T}$ and $\mathcal{G}$ using the proposed C-StarGAN model.
- Appendix D: Details of the experimental settings.

Table 5: **In-domain** test accuracies on PACS dataset, where 75% of all domains samples were used for training and the rest of 25% for test. The best (green), the second best (blue).

Algorithm	Art-Painting	Cartoon	Photo	Sketch	Avg
ERM	91.5 ± 0.6	94.3 ± 0.4	96.0 ± 0.4	95.3 ± 0.4	94.27
DRO	92.7 ± 0.4	94.6 ± 0.3	94.5 ± 0.3	94.9 ± 0.4	94.17
Mixup	93.6 ± 0.4	94.7 ± 0.1	97.0 ± 0.8	96.0 ± 0.3	95.32
MLDG	92.6 ± 0.6	94.5 ± 0.3	96.2 ± 0.8	95.0 ± 0.5	94.57
CORAL	92.5 ± 0.3	94.3 ± 0.5	96.4 ± 0.8	95.5 ± 0.4	94.67
MMD	92.1 ± 0.4	93.6 ± 0.3	95.7 ± 0.7	94.8 ± 0.5	94.05
MTL	93.0 ± 0.3	95.0 ± 0.6	95.5 ± 0.5	93.3 ± 0.6	94.20
SagNet	93.2 ± 0.4	93.4 ± 0.8	95.4 ± 0.9	93.4 ± 0.8	93.85
DIR (Ours)	94.3 ± 0.5	95.5 ± 0.3	96.9 ± 0.3	96.3 ± 0.4	95.75

Table 6: **In-domain** test accuracies on PACS dataset, where 50% of all domains samples were used for training and the rest of 50% for test. The best (green), the second best (blue).

Algorithm	Art-Painting	Cartoon	Photo	Sketch	Avg
ERM	91.4 ± 0.5	93.2 ± 0.5	95.6 ± 0.4	94.3 ± 0.2	93.62
DRO	91.6 ± 0.4	93.7 ± 0.3	95.9 ± 0.3	95.0 ± 0.2	94.05
Mixup	91.5 ± 0.5	94.3 ± 0.1	96.5 ± 0.8	95.4 ± 0.1	94.42
MLDG	91.8 ± 0.6	92.7 ± 0.1	95.4 ± 0.8	95.2 ± 0.5	93.77
CORAL	92.7 ± 0.4	93.8 ± 0.5	96.3 ± 0.5	95.2 ± 0.5	94.65
MMD	90.9 ± 0.2	92.6 ± 0.4	96.6 ± 0.4	94.5 ± 0.5	93.65
MTL	92.8 ± 0.3	94.6 ± 0.6	95.2 ± 0.5	93.3 ± 0.6	93.97
SagNet	93.0 ± 0.4	93.0 ± 0.8	95.1 ± 0.9	93.1 ± 0.8	93.55
DIR (Ours)	93.6 ± 0.5	95.2 ± 0.4	96.4 ± 0.4	96.3 ± 0.4	95.37

B In-Domain Generalization Results

In order to evaluate the proposed framework for in-domain generalization setting (the test set contains a mix of unseen samples from source domains), we conducted two experiments, in which we split the source domain samples into training and test sets with various proportions. Then, we trained a model on source training samples and tested it on source test samples. The first experiment was done by using 75% of source samples as training and the rest of 25% as test samples. The second experiment was done by using 50% of source samples as training and the rest of 50% as test samples (Results are shown in Tabs. 5, 6, 7, 8, 9, and 10. We can see that the **DIR** achieves the best average accuracy on all PACS, OfficeHome, and mini-DomainNet datasets, respectively. **DIR** outperforms the baseline (**ERM**) that aggregates all source domains to train a single model by 1.7%, 4.5%, and 3.8% for PACS, OfficeHome, and mini-DomainNet, respectively.

C Learning $\mathcal{F}, \mathcal{T}$ and $\mathcal{G}$ using C-StarGAN

we can use several I2I models to learn $\mathcal{F}, \mathcal{T}$ and $\mathcal{G}$. In particular, we use the StarGAN (Choi et al., 2020) model, which is designed for multiple image domain transformations. The StarGAN contains four module namely, an image generator module, a style generator module, a style mapping module, and a domain discriminator module. We consider $\mathcal{T}$, and $\mathcal{G}(\mathcal{F}(.))$ as the style generator module and the image generator module of the StarGAN, respectively. Given an image $\mathbf{x}$ and its domain label $\mathbf{d}$, four important objective functions of the StarGAN model to train $\mathcal{F}, \mathcal{T}, \mathcal{G}$ are:

- **Domain Adversarial objective**: During training, a latent code $\mathbf{u} \sim \mathcal{N}(0, I)$ and a target domain $\mathbf{d}$ are randomly sampled to generate a target (domain-specific) style code $\tilde{\mathbf{z}}_s = \mathcal{T}(\mathbf{u}, \mathbf{d})$. $\mathcal{F}$ and $\mathcal{G}$ are learned to generate an output image $\tilde{\mathbf{x}} = \mathcal{G}(\mathcal{F}(\mathbf{x}), \tilde{\mathbf{z}}_\mathbf{s})$ via an adversarial loss using a domain discriminator D. $\mathcal{T}$ learns to provide the style code $\tilde{\mathbf{z}}_s$

Table 7: **In-domain** test accuracies on OfficeHome dataset, where 75% of all domains samples were used for training and the rest of 25% for test. The best (green), the second best (blue).

Algorithm	Art	ClipArt	Product	Photo	Avg
ERM	68.4 ± 0.6	74.7 ± 0.4	86.1 ± 0.4	75.5 ± 0.4	76.17
DRO	66.7 ± 0.6	75.7 ± 0.3	85.6 ± 0.4	77.8 ± 0.4	76.45
Mixup	69.7 ± 0.4	77.5 ± 0.3	87.6 ± 0.6	78.6 ± 0.3	78.35
MLDG	68.9 ± 0.7	75.7 ± 0.3	86.7 ± 1.2	78.0 ± 0.1	77.32
CORAL	71.0 ± 0.6	78.4 ± 0.4	87.6 ± 0.3	79.3 ± 0.4	79.07
MMD	68.8 ± 0.3	74.9 ± 0.5	86.7 ± 0.5	77.6 ± 0.9	77.00
MTL	66.1 ± 0.5	76.5 ± 0.5	85.9 ± 0.1	77.8 ± 0.4	76.55
SagNet	68.6 ± 0.1	78.8 ± 0.3	87.0 ± 0.3	79.1 ± 0.2	78.37
DIR (Ours)	71.2 ± 0.4	80.5 ± 0.4	88.8 ± 0.5	80.9 ± 0.5	80.35

that is likely in the target domain $\mathbf{d}$, and $\mathcal{G}$ is learned to utilize $\tilde{\mathbf{z}}_s$ and generate an image that is indistinguishable from real images of the domain $\mathbf{d}$. The domain adversarial loss is formulated as

$$\mathcal{L}_{adv} = \mathbb{E}_{\mathbf{x},\mathbf{d}}[\log D_{\mathbf{d}}(\mathbf{x})] + \mathbb{E}_{\mathbf{x},\tilde{\mathbf{d}},\mathbf{u}}[\log(1 - D_{\tilde{\mathbf{d}}}(\tilde{\mathbf{x}}))], \;\; \tilde{\mathbf{x}} = \mathcal{G}(\mathcal{F}(\mathbf{x}), \mathcal{T}(\mathbf{u}, \tilde{\mathbf{d}})), \tag{9}$$

where $D_{\mathbf{d}}(.)$ denotes the output of the discriminator D corresponding to the domain $\mathbf{d}$.

- **Style reconstruction Objective**. In order to enforce the generator module to utilize the style code $\tilde{\mathbf{z}}_s$ when generating the image $\tilde{\mathbf{x}}$, a style reconstruction loss is employed to learn a mapping E from an image to its style code. The style reconstruction loss can be written as

$$\mathcal{L}_{sty} = \mathbb{E}_{\mathbf{x},\tilde{\mathbf{d}},\mathbf{u}}\big[||\tilde{\mathbf{z}}_s - E_{\tilde{\mathbf{d}}}(\tilde{\mathbf{x}})||_1\big], \;\; \tilde{\mathbf{x}} = \mathcal{G}(\mathcal{F}(\mathbf{x}), \tilde{\mathbf{z}}_{\mathbf{s}}), \;\; \tilde{\mathbf{z}}_{\mathbf{s}} = \mathcal{T}(\mathbf{u}, \tilde{\mathbf{d}}), \tag{10}$$

where $E_{\mathbf{d}}(.)$ denotes the output of the mapping network E corresponding to the domain $\mathbf{d}$.

- **Style diversification Objective**: To enable the generator module to produce diverse images, $\mathcal{G}$ and $\mathcal{F}$ are regularized with a diversity sensitive loss. The regularization term forces $\mathcal{G}$ and $\mathcal{F}$ to explore the image space and discover meaningful style features to generate diverse images. The style diversification loss can be expressed as

$$\mathcal{L}_{sd} = \mathbb{E}_{\mathbf{x},\tilde{\mathbf{d}},\mathbf{u}_1,\mathbf{u}_2}\big[||\mathcal{G}(\mathcal{F}(\mathbf{x}), \mathbf{z}_1) - \mathcal{G}(\mathcal{F}(\mathbf{x}), \mathbf{z}_2)||_1\big], \;\; \mathbf{z}_1 = \mathcal{T}(\mathbf{u}_1, \tilde{\mathbf{d}}), \;\; \mathbf{z}_2 = \mathcal{T}(\mathbf{u_2}, \tilde{\mathbf{d}}), \tag{11}$$

- **Cycle Consistency Objective**: To guarantee that the generated image $\tilde{\mathbf{x}}$ properly preserves the domain invariant characteristics (e.g. shape) of its input image $\mathbf{x}$, a cycle consistency loss is used. This loss encourages the generator module to preserve the original characteristics of $\mathbf{x}$ while changing its style faithfully. This loss can be represented as

$$\mathcal{L}_{cyc} = \mathbb{E}_{\mathbf{x},\mathbf{d},\tilde{\mathbf{d}},\mathbf{u}}\big[||\mathbf{x} - \mathcal{G}(\mathcal{F}(\tilde{\mathbf{x}}), \mathbf{s})||_1\big], \;\; \tilde{\mathbf{x}} = \mathcal{G}(\mathcal{F}(\mathbf{x}), \mathcal{T}(\mathbf{u}, \tilde{\mathbf{d}})), \;\; \mathbf{s} = E_d(\mathbf{x}), \tag{12}$$

Category Classification Objective: In order to leverage the category labels of source samples, we propose to incorporate a classification module $\mathcal{C} : \mathcal{X} \rightarrow \mathcal{Y}$ into the StarGAN model. Hence, we propose C-starGAN by adding a classification loss function $\mathcal{L}_{class}$ into the StarGAN:

$$\mathcal{L}_{class} = \mathbb{E}_{(\mathbf{x},\mathbf{y})\sim\mathbb{P}(\mathbf{x},\mathbf{y})}\big[-\log([\mathcal{C}(\mathbf{x})]_y)\big], \tag{13}$$

Full objective. The full objective functions of the C-StarGAN can be summarized as

$$\min_{\mathcal{G},\mathcal{F},\mathcal{T},\mathcal{C}} \max_{D} \;\; \mathcal{L}_{adv} + \lambda_{sty}\mathcal{L}_{sty} + \lambda_{cyc}\mathcal{L}_{cyc} - \lambda_{ds}\mathcal{L}_{ds} + \lambda_{class}\mathcal{L}_{class}, \tag{14}$$

where $\lambda_{sty}, \lambda_{cyc}, \lambda_{ds}$, and λ_{class} are hyperparameters for each term. It should be noted that, during the training the C-StarGAN, the classifier $\mathcal{C}$ is only trained on the actual labeled source samples. On the other hand, $\mathcal{G}$ and $\mathcal{F}$ are trained using the classification loss of generated samples as well as other StarGAN loss functions.

Table 8: **In-domain** test accuracies on OfficeHome dataset, where 50% of all domains samples were used for training and the rest of 50% for test. The best (green), the second best (blue).

Algorithm	Art	ClipArt	Product	Photo	Avg
ERM	61.5 ± 0.5	71.8 ± 0.5	82.6 ± 0.5	72.4 ± 0.4	72.07
DRO	63.0 ± 0.5	72.8 ± 0.5	82.8 ± 0.5	72.6 ± 0.2	72.80
Mixup	63.6 ± 0.5	73.5 ± 0.4	85.3 ± 0.6	72.7 ± 0.4	73.77
MLDG	62.8 ± 0.7	72.4 ± 0.5	83.7 ± 0.2	73.2 ± 0.1	73.02
CORAL	66.1 ± 0.6	75.6 ± 0.2	85.8 ± 1.3	75.6 ± 0.2	75.77
MMD	60.5 ± 0.3	72.2 ± 0.5	83.6 ± 0.5	73.2 ± 0.6	72.37
MTL	61.7 ± 0.5	73.6 ± 0.5	83.9 ± 0.1	74.9 ± 0.4	73.52
SagNet	65.2 ± 0.1	75.9 ± 0.3	86.1 ± 0.3	74.7 ± 0.2	75.47
DIR (Ours)	67.1 ± 0.4	77.8 ± 0.4	87.1 ± 0.4	76.6 ± 0.4	77.15

Table 9: **In-domain** test accuracies on mini-DomainNet dataset, where 75% of all domains samples were used for training and the rest of 25% for test. The best (green), the second best (blue).

Algorithm	ClipArt	Painting	Real	Sketch	Avg
ERM	75.4 ± 0.4	71.6 ± 0.5	78.5 ± 0.2	73.7 ± 0.1	74.80
DRO	76.5 ± 0.4	72.0 ± 0.4	80.0 ± 0.5	73.5 ± 0.1	75.50
Mixup	77.8 ± 0.2	73.4 ± 0.5	79.8 ± 0.3	75.3 ± 0.3	76.57
MLDG	77.0 ± 0.2	71.5 ± 0.2	78.7 ± 0.3	73.6 ± 0.1	75.20
CORAL	77.9 ± 0.2	73.6 ± 0.4	80.4 ± 0.6	75.2 ± 0.1	76.77
MMD	75.9 ± 0.5	71.9 ± 0.2	79.0 ± 0.2	74.5 ± 0.7	75.32
MTL	75.4 ± 0.2	73.5 ± 0.4	80.6 ± 0.6	75.3 ± 0.1	76.20
SagNet	75.0 ± 0.2	73.0 ± 0.4	80.7 ± 0.6	75.0 ± 0.1	75.92
DIR (Ours)	79.3 ± 0.3	74.8 ± 0.3	81.6 ± 0.2	76.9 ± 0.1	78.15

D EXPERIMENTAL SETTINGS

In this section, we provide further experimental details beyond the results presented in the main paper. The experiments on all datasets were performed using the DomainBed package. All of the default hyperparameters (e.g. batch-size, learning rate, weight decay, etc.) were left unchanged from the standard DomainBed implementation. We also set the hyper-parameter $M = 1$ (the number of perturbed samples for each source sample).

C-StarGAN Experimental Setting: The StarGAN (Choi et al., 2020) model implementation is taken from the authors' original source code with no significant modifications. All of the default

Table 10: **In-domain** test accuracies on mini-DomainNet dataset, where 50% of all domains samples were used for training and the rest of 50% for test. The best (green), the second best (blue).

Algorithm	ClipArt	Painting	Real	Sketch	Avg
ERM	73.0 ± 0.5	68.4 ± 0.5	76.4 ± 0.5	72.0 ± 0.5	72.45
DRO	73.5 ± 0.4	70.0 ± 0.4	77.6 ± 0.5	71.6 ± 0.1	73.15
Mixup	75.6 ± 0.5	71.2 ± 0.4	78.5 ± 0.4	73.5 ± 0.4	74.70
MLDG	73.6 ± 0.2	70.1 ± 0.4	76.5 ± 0.4	71.4 ± 0.3	73.01
CORAL	76.4 ± 0.5	70.0 ± 0.5	78.0 ± 0.6	73.0 ± 0.3	74.35
MMD	75.4 ± 0.4	68.5 ± 0.4	77.5 ± 0.5	71.5 ± 0.5	73.68
MTL	73.0 ± 0.2	70.5 ± 0.4	78.6 ± 0.6	73.5 ± 0.1	73.90
SagNet	73.0 ± 0.2	71.1 ± 0.4	78.9 ± 0.6	73.2 ± 0.1	74.05
DIR (Ours)	78.5 ± 0.3	72.3 ± 0.5	79.7 ± 0.5	75.7 ± 0.5	76.55

Algorithm 1 Domain Invariant Regularization (DIR) for Domain Generalization

Hyperparameters: step size η, number of perturbations M, mini-batch size B, Trade-off parameter λ, Number of source domains S
Parameters: The Classifier f Parameters (θ)
C-StarGAN Training: Learn $\mathcal{F}, \mathcal{G}, and \mathcal{T}$ using the C-StarGAN objective function in Eq. 14.
repeat
  for minibatch $\{(\mathbf{x}_i, \mathbf{y}_i, \mathbf{d}_i)\}_{i=1}^{B}$ in training dataset $D_1 \cup D_2 \cup \cdots \cup D_S$ **do**

    $\{\tilde{\mathbf{x}}_j\}_{j=1}^{M} \leftarrow \text{PERTURBIMAGE}(\mathbf{x}_i)\ \forall i \in [B]$ ▷ Generate M perturbed samples

    $\mathcal{L}_{reg}(\theta) \leftarrow (\frac{1}{BM}) \sum_{i=1}^{b} \sum_{j=1}^{M} ||(f_\theta(\mathbf{x}_i) - f_\theta(\tilde{\mathbf{x}}_i)||_1$ ▷ Calculate regularizer in Eq. 7

    $\mathcal{L}_{cls}(\theta) \leftarrow (-\frac{1}{B}) \sum_{i=1}^{B} \log([f(\mathbf{x}_i)]_{\mathbf{y}_i})$ ▷ Calculate classification loss in Eq. 7

    $\theta \leftarrow \theta - \eta \nabla_\theta [\, \mathcal{L}_{cls}(\theta) + \lambda \cdot \mathcal{L}_{reg}(\theta)\,]$ ▷ Gradient step for θ
  end for
until convergence

procedure PERTURBIMAGE($\mathbf{x}$)
  $\mathbf{z}_c \leftarrow F(x)$ ▷ Extract the content feature $\mathbf{z}_c$ of $\mathbf{x}$
  Sample $\mathbf{u} \sim \mathcal{N}(0, I)$
  Sample $\tilde{\mathbf{d}} \sim U(\{1, 2, ..., S\})$
  $\tilde{\mathbf{z}}_s \leftarrow \mathcal{T}(\mathbf{u}, \tilde{\mathbf{d}})$ ▷ Generate a random style feature $\tilde{\mathbf{z}}_s$ in domain $\tilde{\mathbf{d}}$
  return $\mathcal{G}(\mathbf{z}_c, \tilde{\mathbf{z}}_s)$ ▷ Return perturbed image produced by C-StarGAN
end procedure

hyperparameters (e.g. λ_{cyc}, λ_{sty}, λ_{ds}, batch-size, learning rate, weight decay, model architectures, etc.) were left unchanged from the StarGAN implementation. We define the architecture for the classification module $\mathcal{C}$ the same as the StarGAN discriminator module D except for the last layer, where we use a single linear layer instead of multiple linear layers. The details of the classification module is shown in Tab. 11. For each set of source domains, we train the StarGAN model for $150,000$ iterations. We also set λ_{class} to 0.1 for all datasets.

LAYER	RESAMPLE	NORM	OUTPUT SHAPE
Image $\mathbf{x}$	-	-	256×256×3
Conv1×1	-	-	256×256×64
ResBlk	AvgPool	-	128×128×128
ResBlk	AvgPool	-	64×64×256
ResBlk	AvgPool	-	32×32×512
ResBlk	AvgPool	-	16×16×512
ResBlk	AvgPool	-	8×8×512
ResBlk	AvgPool	-	4×4×512
LReLU	-	-	4×4×512
Conv4×4	-	-	1×1×512
LReLU	-	-	1×1×512
Reshape	-	-	512
Linear	-	-	1 ⋆ K

Table 11: C-StarGAN classification module architecture. K represent the number of category labels.