Domain-Conditioned Normalization for Test-Time Domain Generalization

Yuxuan Jiang[1], Yanfeng Wang[1,2], Ruipeng Zhang[1], Qinwei Xu[1], Ya Zhang[1,2], Xin Chen[3], and Qi Tian[4]

[1] Cooperative Medianet Innovation Center, Shanghai Jiao Tong University
[2] Shanghai AI Laboratory
[3] Huawei Inc.
[4] Huawei Cloud & AI
{g.e.m_jyx, wangyanfeng, zhangrp, qinweixu, ya_zhang}@sjtu.edu.cn, chenxin061@gmail.com, tian.qi1@huawei.com

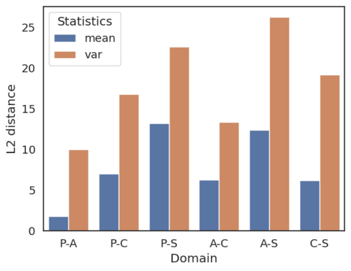
Abstract. Domain generalization aims to train a robust model on multiple source domains that generalizes well to unseen target domains. While considerable attention has focused on training domain generalizable models, a few recent studies have shifted the attention to test time, *i.e.,* leveraging test samples for better target generalization. To this end, this paper proposes a novel test-time domain generalization method, Domain Conditioned Normalization (DCN), to infer the normalization statistics of the target domain from only a single test sample. In order to learn to predict the normalization statistics, DCN adopts a meta-learning framework and simulates the inference process of the normalization statistics at training. Extensive experimental results have shown that DCN brings consistent improvements to many state-of-the-art domain generalization methods on three widely adopted benchmarks.

Keywords: domain generalization, batch normalization, test time adaptive, distribution shift

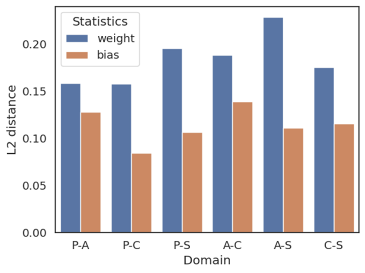
1 Introduction

The performance of deep neural networks degrades drastically when the distribution of train (source) and test(target) data are different. In order to solve the problem of distribution shift, domain generalization(DG) aims to train a robust model on multiple source domains that generalizes well to arbitrary unseen target domains. Existing DG methods can be mainly divided into three categories: invariant representation learning which aims to learn a shared feature space across source domains, data augmentation which expands the source distributions, regularization techniques which are utilized to learn more semantic information from source domains.

Despite their success, most existing DG methods only focus on the training stage, *i.e.,* how to train a generalizable model based on source domains. In fact, DG is naturally an ill-posed problem due to information incompleteness. Knowledge of the target domain may greatly help the model generalization. Actually,



(a) Standardization statistics



(b) Rescaling statistics

Fig. 1. The difference in the first BN layer for (a) standardization statistics and (b) rescaling statistics. We train one model on all domains of PACS benchmark with different BN layers for each domain. The L2 distance is used to measure the statistics differences between every two domains.

the model always has access to at least one unlabeled test sample at inference which may provide important clues about the target domain. Some recent studies start to investigate how to make use of the test sample at inference to improve a model's generalizability to the target domain. TTT [33] and TENT [37] utilize target domain data to update model parameters by constructing an unsupervised loss. Dubey et al. [7] use some target samples to construct domain embeddings so that the classifier can make dynamic predictions based on it. T3A [13] computes pseudo-prototype representation for each class with past predicted target samples to modify the classifier. However, these methods have the following two limitations. (1) Updating parameters with target data increases the inference time significantly and may lead to catastrophic failure; (2) Batches of data are not necessarily available at inference and there is no guarantee that the domain labels of test samples are available.

In this paper, we propose a test-time DG method called Domain Conditioned Normalization (DCN), which estimates the normalization statistics for target domain at inference. DCN is based on the observation that the statistics (including standardization and rescaling statistics) in batch normalization(BN) [12] are unique for each domain and larger distribution shift is reflected by the increased difference in statistics, as shown in Fig. 1. Thus, to improve the domain generalizability of a model at test time, a straight-forward solution is to normalize the test data with the statistics of the domain it comes from. The challenge lies in we can assume access to only one test sample. Nevertheless, the good news is that, the source statistics are expected to contain essential semantic and style information shared by the source and target domains. We thus attempt to combine the instance statistics from a single test sample that reflect domain-specific information for the target domain, with the source statistics to infer the statistics of the target domain. The final inferred statistics is then used to normalize this test sample during inference.

In order to infer the target statistics, a meta-learning framework is adopted by DCN. In each meta-iteration, one source domain is chosen as the meta-target domain and the other source domains are considered as meta-source domains. DCN learns to infer the statistics of the meta-target domain by approximating the ground-truth meta-target statistics with the instance statistics of the meta-target sample and the statistics of meta-source domains. At testing, DCN infers the statistics of the target domain with a single test sample and the source statistics, which is used to replace the source statistics in the trained model for prediction. Our DCN only requires one forward pass of a single target sample, which avoids training at test time and requires no batched test data.

To validate the effectiveness of the proposed method, we experiment on several standard DG benchmarks, namely PACS [17], VLCS [35], OfficeHome [36]. Extensive experimental results have shown that DCN brings consistent improvements to the state-of-the-art DG approaches, indicating that the inferred target statistics obtained from a single test sample does help model generalize better across domains. We have also demonstrated that DCN becomes more effective as the distribution shift increases.

2 Related Work

2.1 Domain Generalization

The main goal of domain generalization(DG) is to learn a robust model from multiple available source domains that can generalize well to arbitrary unseen target domains [38,44]. Existing DG methods can be roughly categorized into three classes: invariant representation learning, data augmentation and regularization methods. Invariant representation learning aims to learn a shared feature space that retains the semantic information across source domains. Some work [19,22] learn this shared feature space in an adversarial training manner. Another work [32,31,29] align second-order statistics or gradients to learn the invariant representation. Data augmentation is also a common method for DG research. DDAIG [45] and FACT [41] generate new images to enlarge the training dataset. Different from augmentation at the image level, some methods [46,20,47] expand source distributions at the feature level. Another popular approach in DG is to use regularization techniques. Self-supervised objective is a popular choice as the regularization [2,39,16,4]. Another work [18,1] rely on meta-learning to simulate the domain shift during training. Furthermore, RSC [11] masks the maximum response gradient to learn more semantic information. These approaches only focus on source domains at the training staget and ignores the use of test data during inference.

2.2 Normalization in Neural Networks

Batch Normalization [12] is a common technique that optimizes the training of deep network by reducing internal covariate shift. However, normalizing the target domain data with the source statistics is sub-optimal because there is a distribution shift between source and target domains. To solve this problem, BIN [27] combines different standardization statistics with the learnable weights. ILM [14] learns the rescaling statistics for each sample by using neural networks. DSON [28] proposes to learn separate BIN for each source domain and ensemble the normalization for the target domain. MetaNorm [6] and ASR-Norm [8] learn the adaptive statistics based on a single sample. Different from all the methods above, our DCN uses both a single target sample and source statistics for adaptive normalization, so DCN can infer more accurate target statistics and achieve better generalization performance.

2.3 Adaptation and Generalization at Test Time

Recently, a lot of work has begun to study test-time adaptation. TTT [33] and TENT [37] update model parameters by training the target domain data with an unsupervised loss function. Apart from that, some work improves model generalization with test data while avoiding training at test time. BN-Test [26] computes statistics on batches of target samples to normalize them during inference. Dubey et al. [7] use very few test samples to construct the domain embedding for the target domain and use the domain embedding as a supplementary signal to make adaptive predictions. T3A [13] computes the representation templates with previous target samples and use these representation templates to modify the classifier. Compared with these methods, our DCN only requires a single test sample and avoids training during inference so that it avoids the increase of inference time and memory overhead.

3 Methodology

We here attempt to explore adaptive normalization at test time by inferring the target domain statistics with source domain statistics and a single target sample. In this section, we first describe the formal formulation of domain generalization and domain-specific batch normalization, followed by the details of our method, Domain Conditioned Normalization (DCN).

3.1 Preliminary

Domain Generalization In domain generalization, the training dataset D_S consists of N_S source domains. Each source domain $D_d = \{(x_d^i, y_d^i)\}_{i=1}^{n_d}$ $(d \in \{1, \cdots, N_S\})$ contains n_d data and label pairs. There is also a target domain D_T, which is only available during testing. The goal of domain generalization is to train a model f_θ with D_S that generalizes well on the target domain D_T, where θ is the parameters of the model. A vanilla baseline is training with all source domain data with the empirical risk minimization (ERM) objective:

$$\min_{\theta} \frac{1}{N_S} \sum_{d=1}^{N_S} \frac{1}{n_d} \sum_{i=1}^{n_d} l(f_\theta(x_d^i), y_d^i), \tag{1}$$

where $l(\cdot)$ is the standard cross-entropy loss function for classification problems.

Domain Specific Batch Normalization BN [12] is a widely used training technique in deep learning to reduce the internal covariate shift. The statistics in BN represent the characteristics of the training dataset. However, when present with multiple training domains, it is inappropriate to share the BN statistics among different domains considering each domain have its own characteristics. Therefore, in this paper, we employ DSBN [3] to separately store the domain-specific statistics for each source domain, which makes data from different domains go through different BN layers.

Given a batch of samples from domain d during training, the input feature maps of a normalization layer is denoted by $x_d \in \mathbb{R}^{N\times C\times H\times W}$, where N denotes the batch size, C denotes the number of channels, H and W denote height and width respectively. DSBN first uses the domain-specific batch mean μ_d^{bn} and variance σ_d^{bn} for standardization, then applies affine transformations on the standardized features x_d^{stan} with domain-specific rescaling parameters $\{\gamma_d,\ \beta_d\}$. The whole process is denoted as:

$$x_d^{\text{stan}} = \frac{x_d - \mu_d^{\text{bn}}}{\sqrt{\sigma_d^{\text{bn}} + \epsilon}}, \tag{2}$$

$$x_d^{\text{dsbn}} = \gamma_d \cdot x_d^{\text{stan}} + \beta_d, \tag{3}$$

where $\mu_d^{\text{bn}}, \sigma_d^{\text{bn}}, \gamma_d, \beta_d \in \mathbb{R}^C$ and ϵ is a small constant for numerical stability. The batch mean μ_d^{bn} and variance σ_d^{bn} are calculated as:

$$\mu_d^{\text{bn}} = \frac{\sum_{n,h,w} x_d}{N\cdot H\cdot W} \quad \text{and} \quad \sigma_d^{\text{bn}} = \frac{\sum_{n,h,w}(x_d - \mu_d^{\text{bn}})^2}{N\cdot H\cdot W}. \tag{4}$$

Following standard implementations [12], we acquire the domain-specific mean μ_d and variance σ_d of a source domain D_d through Exponential Moving Average (EMA). With the help of DSBN, we obtain the domain-specific normalization statistics (*i.e.*, standardization statistics $\{\mu_d,\ \sigma_d\}$ and rescaling statistics $\{\gamma_d,\ \beta_d\}$) of different source domains, which facilitates the training of DCN.

3.2 Domain Conditioned Normalization

Our approach aims to use a single target sample and source domain-specific statistics to infer the normalization statistics for an unseen target domain. To accomplish this, we adopt a meta-learning strategy to simulate such a target statistics inference task during each meta-iteration.

Specifically, within a meta-iteration, we randomly split the training domains D_S into two disjoint sets: one domain as meta-target and the others are meta-source domains. In the meta-train stage, we train the backbone network on meta-source domains and update the DSBN statistics of each meta-source domain by minimizing the cross-entropy loss. In the meta-test stage, we update the DSBN statistics of the meta-target domain in the same way to provide ground-truth meta-target statistics. Then we train the DCN modules which take the

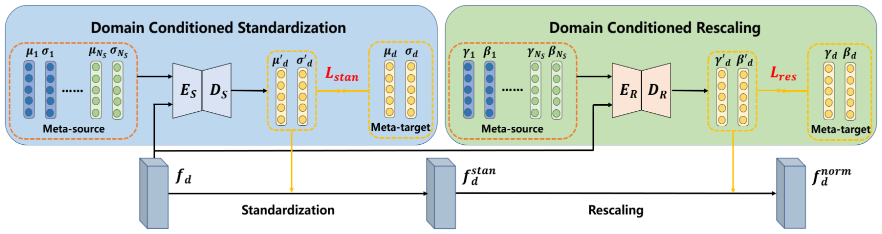


Fig. 2. Illustration of DCN for each layer when D_d is the meta-target domain. We use auto-encoders to infer statistics of D_d with input feature map f_d and the meta-source domains' statistics and use the inferred statistics to normalize f_d. L_{stan} and L_{res} are L2 loss between the inferred statistics and the ground truth statistics.

statistics of the meta-source domains and a single meta-target sample as inputs, to approximate the ground-truth meta-target statistics.

By looping over multiple meta-iterations, we hope the model would be able to infer the DSBN statistics on a novel target domain during testing. The learning process of DCN for a single layer is illustrated in Fig. 2, which can be further divided into domain conditioned standardization and domain conditioned rescaling. Next, we will describe these two parts in detail.

Domain Conditioned Standardization The standardization statistics stand for the channel-wise mean μ and variance σ. To infer the meta-target standardization statistics, we adopt an auto-encoder structure, where both the encoder and decoder are composed of one fully-connected layer, followed by a ReLU activation. The predictions of the standardization auto-encoder are conditioned on the instance mean μ_d^i and variance σ_d^i of the meta-target sample x_d^i, as well as the standardization statistics $\{\mu_p, \sigma_p\}$ $(p \neq d,\ p \in \{1, \cdots, N_S\})$ of the meta-source domains. The instance mean μ_d^i and variance σ_d^i of x_d^i are calculated as:

$$\mu_d^i = \frac{\sum_{h,w} x_d^i}{H \cdot W} \quad \text{and} \quad \sigma_d^i = \frac{\sum_{h,w}(x_d^i - \mu_d^i)^2}{H \cdot W}. \tag{5}$$

Inspired from [15,42], we assume the standardization statistics of one domain to be a shifted version of those of another domain. Therefore, the meta-target standardization statistics can be inferred in the form of a linear combination of the meta-source standardization statistics and the instance statistics of x_d^i, which is specified as:

$$\mu'_d = \frac{1}{N_S - 1} \sum_{p \neq d} (w_{p,d}^i \cdot \mu_d^i + (1 - w_{p,d}^i) \cdot \mu_p), \tag{6}$$

$$\sigma'_d = \frac{1}{N_S - 1} \sum_{p \neq d} (w_{p,d}^i \cdot \sigma_d^i + (1 - w_{p,d}^i) \cdot \sigma_p). \tag{7}$$

where $w_{p,d}^i \in \mathbb{R}^C$ is the channel-wise linear combination weights learned from the standardization auto-encoders. Intuitively, different channels correspond to different degrees of transferability from the meta-source D_p to the meta-target D_d, so using a specific weight for each channel is a wiser choice.

To learn the combination weight $w^i_{p,d}$, we adopt a similar strategy as [21,40] by using the difference between the instance statistics $\{\mu^i_d,\ \sigma^i_d\}$ of x^i_d and the meta-source standardization statistics $\{\mu_p,\ \sigma_p\}$ as the inputs to the auto-encoder. The difference metric is calculated by:

$$M^i_{p,d} = \left| \frac{\mu_p}{\sqrt{\sigma_p + \epsilon}} - \frac{\mu^i_d}{\sqrt{\sigma^i_d + \epsilon}} \right| . \tag{8}$$

Intuitively, the channel-wise statistics difference $M^i_{p,d}$ can be viewed as an indicator of the domain transferability, where a small value of an entry in $M^i_{p,d}$ means the meta-source domain and the meta-target domain are similar in the corresponding channel, and vice versa. Such a transferability indicator allows the auto-encoder to learn a better trade-off between the meta-source standardization statistics and the instance statistics of x^i_d. Finally, the output of the decoder goes through a sigmoid activation to scale the learned weights into [0, 1].

After obtaining the inferred standardization statistics of D_d through Eq. (6) and Eq. (7), we train the standardization auto-encoder by minimizing the L2 distance with the ground-truth standardization statistics $\{\mu_d,\ \sigma_d\}$ obtained from DSBN on the meta-target domain:

$$L_{stan} = ||\mu'_d - \mu_d||^2_2 + ||\sigma'_d - \sigma_d||^2_2. \tag{9}$$

Domain Conditioned Rescaling The rescaling statistics stand for the channel-wise affine parameters $\{\gamma,\ \beta\}$. Denote the rescaling statistics of the meta-target domain D_d and the meta-source domain D_p as $\{\gamma_d,\ \beta_d\}$ and $\{\gamma_p,\ \beta_p\}$, respectively. Similar as domain conditioned standardization, our goal is to learn a rescaling auto-encoder to infer the rescaling statistics of the meta-target domain D_d. The rescaling auto-encoder shares the same structure as the standardization auto-encoder, while its predictions are conditioned on the instance statistics $\{\mu^i_d,\ \sigma^i_d\}$ of the single meta-target sample x^i_d and the rescaling statistics $\{\gamma_p,\ \beta_p\}$ of the meta-source domain.

As stated in [10,34,23], the rescaling statistics $\{\gamma,\ \beta\}$ act like an attention mechanism that measures the contributions of each channel to the current domain. When transferring from the meta-source domain to the meta-target domain, the rescaling statistics of the domain-shared channels should be kept as the same since these channels make similar contributions on both domains. In contrast, the rescaling statistics of the non-shared channels should be calibrated to learn the meta-target domain-specific information. Therefore, we devise the learning strategy of the meta-target rescaling statistics in the manner of a scaling of the meta-source rescaling statistics. Specifically, we concatenate σ^i_d with γ_p and μ^i_d with β_p. Then we feed the two concatenated vectors into the rescaling auto-encoder and infer the meta-target rescaling statistics as:

$$\gamma'_d = \frac{1}{N_S - 1} \sum_{p \neq d} (g_r([\sigma^i_d, \gamma_p]) + 1) \cdot \gamma_p, \tag{10}$$

Algorithm 1 The Meta-learning Strategy for DCN

```
Input: Model $f_\theta$, Total Iterations $T$, Source Domains $D_S$.
Output: Model $f_\theta$, Statistics on $D_S$ ($\{\mu, \sigma\}$, $\{\gamma, \beta\}$).
1: for all $t$ in $1 \cdots T$ do
2:     Start a meta-iteration:
3:         Random split $D_S$ into meta-source domains $D_p$ and meta-target domain $D_d$.
4:         Meta-train:
5:             Forward and infer meta-source DSBN statistics $\{\mu_p, \sigma_p\}$, $\{\gamma_p, \beta_p\}$.
6:         Meta-test:
7:             Random select one sample $x_d^i$ from $D_d$.
8:             Obtain instance statistics $\{\mu_d^i, \sigma_d^i\}$ with Eq. (5).
9:             Infer standardization statistics with Eq. (6) and Eq. (7).
10:            Infer rescaling statistics with Eq. (10) and Eq. (11).
11:            Infer ground-truth meta-target DSBN statistics $\{\mu_d, \sigma_d\}$, $\{\gamma_d, \beta_d\}$.
12:            Compute the total loss with Eq. (13) and backward.
13: end for
```

$$\beta'_d = \frac{1}{N_S - 1} \sum_{p \neq d} (g_r([\mu_d^i, \beta_p]) + 1) \cdot \beta_p. \tag{11}$$

where a Tanh activation is further appended to scale the decoder outputs.

After acquiring the inferred rescaling statistics of the meta-target domain, we minimize its L2 distance with the ground truth $\{\gamma_d, \beta_d\}$ obtained from DSBN on the meta-target domain:

$$L_{res} = ||\gamma'_d - \gamma_d||_2^2 + ||\beta'_d - \beta_d||_2^2. \tag{12}$$

3.3 Training and Inference

Training In our meta-learning process, the DCN modules attempt to approximate the meta-target DSBN statistics conditioned on only a single meta-target sample and the meta-source DSBN statistics. To obtain the meta-source DSBN statistics, we need to train the backbone network during the meta-train stage. Similarly, we should also train the backbone network during the meta-test stage to acquire reliable meta-target DSBN statistics as the ground truth for the approximation. Since the meta-source and meta-target domain are randomly selected in each meta-iteration and the whole training process goes through multiple meta-iterations, we could merge the training of DSBN on both the meta-source and meta-target domain into only on the meta-target domain. In this way, the meta-source DSBN statistics can be obtained with only a forward propagation without additional training, which results in a more simplified and efficient meta-learning process. The training in the meta-test stage is then composed of training the DSBN statistics with the classification loss and training the DCN modules to approximate DSBN with Eq. (9) and Eq. (12). To ensure the discriminality of DCN, we also add a classification loss on the features normalized by DCN. The total objective of the meta-test stage is then formulated as:

$$L = L_{cls} + \lambda(L'_{cls} + \bar{L}_{stan} + \bar{L}_{res}), \tag{13}$$

where L_{cls} and L'_{cls} are standard cross-entropy loss functions for training DSBN and DCN respectively, $\bar{L}_{stan}$ and $\bar{L}_{res}$ are the average of Eq. (9) and Eq. (12) for each DCN layer and λ are the balancing weight which is set as the gradient magnitudes of L_{cls}. The total training process is shown in Algorithm 1.

Inference The inference process only adds a small amount of computation to the normalization operation and can achieve nearly the same speed as the ERM baseline method. When inferring a test example x_t in the target domain D_T, we first infer the normalization statistics of D_T with x_t and the domain-specific statistics of each source domain on the DCN normalization layers. We can obtain the inferred statistics through each source domain and we average them to normalize x_t. Next, we put the feature map of x_t normalized in this way into the classifier and get the final prediction result.

4 Experiments

In this section, we present both the quantitative and qualitative results of our method. Firstly, we describe the datasets and implementation details. Then, we compare our method with state-of-the-art methods to confirm the effectiveness of DCN. Furthermore, we conduct the ablation study to analyze the properties of our method.

4.1 Experiment Setup

Datasets As our evaluation benchmarks. we use PACS [17], VLCS [35] and Office-Home [36]. Following [17], we apply the leave-one-domain-out protocol for all benchmarks, which means we leave one domain as the target domain and the other domains as source domains. For PACS and VLCS, we use the official split to conduct the experiment. For Office-Home, we split the dataset into 90% training set and 10% validation set randomly [17]. We select the best model on the validation set. All the reported results are conducted five times and averaged.

Implementation details For a fair comparison, we follow the implementations of [2,41,8]. We employ ResNet [9] as our backbone in all experiments and use the pretrained protocol in [17]. As for optimization, the backbone is trained with the SGD optimizer with the weight decay of 5e-4; the auto-encoders are trained with the Adam optimizer. We use a batch size of 32 and train the network for 30 epochs. The learning rate is 0.001 on PACS and Office-Home and 5e-5 on VLCS for the backbone, and the same for the auto-encoders except 0.0001 on PACS. Both of the learning rates are decayed by 0.1 at 80% of the total epochs. We also use some augmentations in RandAug [5] and implement the augmentation with probability 0.5, which makes the statistics of each source domain more robust.

Table 1. Comparison with different DG methods (%) using ResNet-18 and ResNet-50 on PACS [17] dataset. The best results are marked as bold. The asterisk means that ASR-Norm must be combined with RSC.

Backbone	Method	Photo	Art	Cartoon	Sketch	Avg.
ResNet-18	ERM	95.12	78.37	75.16	75.41	81.02
	BIN [27]	95.00	82.10	74.10	80.00	82.80
	EISNet [39]	95.93	81.89	76.44	74.33	82.15
	FACT [41]	95.15	85.37	78.38	79.15	84.51
	DSON [28]	95.87	84.67	77.65	82.23	85.11
	RSC [11]	95.99	83.43	80.31	80.85	85.15
	MetaNorm [6]	95.99	85.01	78.63	83.17	85.70
	ASR-Norm * [8]	96.10	84.80	**81.80**	82.60	86.30
	DCN(Ours)	**96.51**	**86.60**	81.47	**83.60**	**87.05**
ResNet-50	ERM	97.64	84.94	76.98	76.75	84.08
	EISNet [39]	97.11	86.64	81.53	78.07	85.84
	DSON [28]	95.99	87.04	80.62	82.90	86.64
	RSC [11]	**97.92**	87.89	82.16	83.85	87.83
	FACT [41]	96.75	**89.63**	81.77	84.46	88.15
	DCN(Ours)	96.82	89.16	**86.09**	**86.15**	**89.56**

For the dimension of auto-encoders in DCN, the encoders and decoders are $(C, C/2)$, $(C/2, C)$ in standardization stage and $(2C, C)$, (C, C) in rescaling stage (C denotes the channel number in that layer). Moreover, the decoder for γ and β are not shared. We take ϵ as 1e-5. For all experiments, only the BN in the first three residual blocks is replaced by DCN.

4.2 Comparison with State-of-the-art Methods

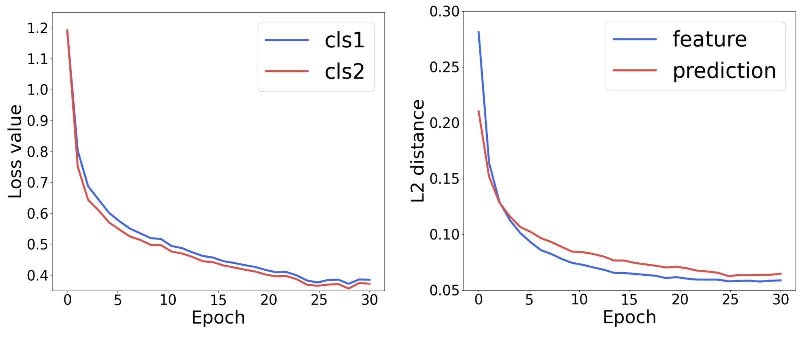
In this section, we evaluate our method on PACS, VLCS and Office-Home. We compare our DCN with recent state-of-the-art DG methods to demonstrate the effectiveness of DCN.

PACS We use ResNet-18 and ResNet-50 as our backbone to perform the evaluation on PACS. We compare with many normalization-based DG methods, such as BIN [27], DSON [28], Meta-Norm [6] and ASR-Norm [8]. We also compare with other SOTA methods, such as EISNet [39], RSC [11] and FACT [41]. The results are shown in Table 1. The ERM baseline can achieve good performance on the photo domain, because the photo domain is similar to the pretrained dataset ImageNet. However, ERM fails on art, cartoon and sketch domains due to the large distribution shift. When inferring target samples, our DCN takes full advantage of the single target sample's hint about the distribution of the target domain. Therefore, the improvement of DCN on ERM is more obvious when the distribution shift is larger.

Compared with the SOTA, our DCN significantly outperforms other DG methods. In ResNet-18, DCN achieves the best performance on three domains and the second best in the other domain, resulting in the best average accuracy.

Table 2. Comparison with different DG methods (%) using ResNet-18 on VLCS [35] dataset. The best performance is marked as **bold.**

Method	Caltech	LabelMe	Pascal	Sun	Avg.
ERM	91.86	61.81	67.48	68.77	72.48
JiGen [2]	96.17	62.06	70.93	71.40	75.14
MMLD [24]	97.01	62.20	73.01	72.49	76.18
RSC [11]	96.21	62.51	73.81	72.10	76.16
StableNet [43]	96.67	**65.36**	73.59	74.97	77.65
DCN(Ours)	**98.23**	62.11	**74.88**	**75.78**	**77.75**



(a) Classification losses (b) Features & predictions

Fig. 3. The analysis of the divergence between the two outputs. (a) compares the classification loss of the two outputs, and (b) compares the L2 distance on both the two feature maps before the classifier and the two prediction vectors. The ablation study is conducted on VLCS dataset with LabelMe as the target domain.

In ResNet-50, DCN has an improvement of 3.93% and 1.69% over the second best on cartoon and sketch domains respectively. Moreover, DCN improves 1.41% over the SOTA method FACT on average accuracy. This results show the effectiveness of DCN when it is incorporated into different backbones.

VLCS The results on VLCS are presented in Table 2. We compare with four DG methods: JiGen [2], MMLD [24], RSC [11] and StableNet [43]. Our method achieves the best performance on three domains and surpasses the latest StableNet [43] in terms of average performance. This shows that DCN can also perform well when the domain shift is small between source and target domains.

Office-Home The results on Office-Home are presented in Table 3. Due to the variations being mainly background and viewpoint, the distribution discrepancy is small on Office-Home, and the best DG method lifts very little on Product and Real-World(0.5% in Product and 0.35% in Real-World). Therefore, ERM acts as a strong baseline and outperforms many DG methods. Our DCN focuses

Table 3. Comparision with different DG methods (%) using ResNet-18 on Office-Home [36] dataset. The best performance is marked as **bold.**

Method	Art	Clipart	Product	Real-World	Avg.
ERM	58.90	49.40	74.30	76.20	64.70
CCSA [25]	59.90	49.90	74.10	75.70	64.90
MMD-AAE [19]	56.50	47.30	72.10	74.80	62.70
CrossGrad [30]	58.40	49.40	73.90	75.80	64.40
JiGen [2]	53.04	47.51	71.47	72.79	61.20
DDAIG [45]	59.20	52.30	74.60	76.00	65.50
L2A-OT [46]	**60.60**	50.10	**74.80**	77.00	65.60
RSC [11]	58.42	47.90	71.63	74.54	63.12
FACT [41]	60.34	54.85	74.48	**76.55**	66.56
DCN(Ours)	59.83	**57.16**	73.78	75.63	**66.60**

Table 4. The ablation study of different components of our method on the PACS dataset with ResNet-18. DCR and DCS represent the model are trained with only the domain conditioned rescaling and domain conditioned standardization components respectively.

Method	Photo	Art	Cartoon	Sketch	Avg.
ERM	95.12	78.37	75.16	75.41	81.02
DCR	95.33	80.35	76.02	76.53	82.06
DCS	96.18	85.65	80.76	82.52	86.28
DCN	**96.51**	**86.60**	**81.47**	**83.60**	**87.05**

on inferring the normalization statistics for the target domain while the spatial features such as viewpoint are not reflected on the normalization statistics, so our method is slightly worse than ERM on Product and Real-World. However, our method improves 7.76% over ERM on Clipart, which is the hardest generalization task. It demonstrates once again that DCN is more effective when the distribution shift between source and target domains is larger. Moreover, DCN achieves comparable performance to the current SOTA method FACT. This again validates the effectiveness of our method.

4.3 Ablation Study

Impact of different components Tabel 4 illustrates the effects of domain conditioned standardization(DCS) and domain conditioned rescaling(DCR) respectively. As shown in Tabel 4, the model only applying DCR has a small improvement on the baseline, while only applying DCS can improve the baseline by 5.26%. This is because the difference in standardization statistics between different domains is much larger than that in rescaling statistics. Furthermore, the model performs best when applying both DCR and DCS(i.e. DCN), which validates the necessity of both modules.

Table 5. The performance comparison of DCN in different locations of the network. The block1-4 means the first to last residual blocks in ResNet-18, respectively. The ablation study is conducted on PACS dataset.

Method	block1	block2	block3	block4	Photo	Art	Cartoon	Sketch	Avg.
Baseline	-	-	-	-	95.12	78.37	75.16	75.41	81.02
Model A	✓	-	-	-	95.65	83.57	77.86	81.09	84.54
Model B	✓	✓	-	-	96.35	85.79	79.86	82.43	86.11
Model C	-	✓	✓	-	96.47	85.55	79.61	81.70	85.83
Model D	✓	✓	✓	-	96.51	**86.6**	**81.47**	**83.6**	**87.05**
Model E	-	✓	✓	✓	96.59	84.03	78.84	81.37	85.21
Model F	✓	✓	✓	✓	**96.77**	84.72	80.33	81.98	85.95

The divergence between the two outputs We analyse the divergence between the two outputs in different aspects. Fig. 3(a) compares the classification loss of the two outputs. The blue line represents L_{cls} and the red line represents L'_{cls}. As shown in Fig. 3(a), the curves of the two classification losses are very close. Fig. 3(b) compares the distance between the two feature maps before the classifier(denoted by the blue line) and the distance between the two prediction vectors(denoted by the red line). We can observe that both distances get smaller and smaller. These two figures demonstrate that the inferred statistics of the meta-target domain are reasonable and get closer to the ground truth statistics of the meta-target domain during training.

4.4 Further Analysis

Effect of where DCN is used We conduct an extensive ablation study to investigate the effect of where DCN is used. As shown is Table 5, only replacing BN with DCN in the first residual block can greatly improve the performance and get 3.52% raise than baseline. As the number of residual blocks using DCN increases, the generalization performance of the model is getting better and better. We can also notice that model F is worse than model D and model E is worse than model C. This is mainly because the normalization statistics in the last residual block are closely related to semantics, so the use of DCN in the last residual block may cause certain disturbances. Meanwhile, model B is better than model C and model D is better than model E. It means the statistics difference between source and target domains in the first residual block is larger, so DCN is more needed in block1 to infer the target statistics instead of using the source statistics during inference.

Weight differences between DCN layers Fig. 4 shows the average weight (which is learned by the standardization auto-encoder) for each DCN layer when cartoon is the target domain on PACS dataset. It is obvious that the shallow DCN layers have a larger average weight than the deep DCN layers for

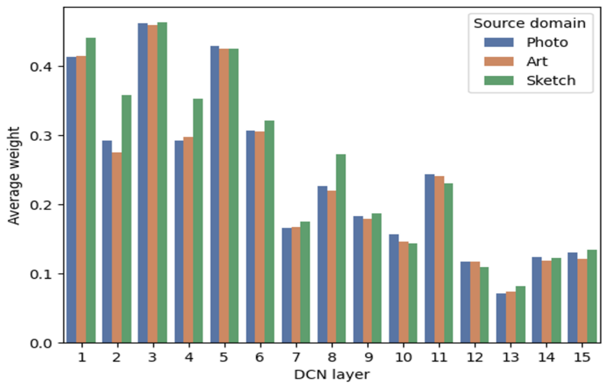


Fig. 4. Analysis of average weights(averaged on each channel) for different DCN layers when cartoon is the target domain on PACS dataset. The backbone is ResNet-18 and we only substitute the first 15 BN with DCN.

all source domains. It confirms the intuition that shallow layers extract richer domain-specific features, so the shallow DCN layers rely more on a single target sample's instance information to explore the domain-specific information of the target domain. On the other hand, deep DCN layers rely more on source statistics because the deep layers extract more semantic features, which are similar between source and target domains.

5 Conclusions

In this paper, we propose a novel domain generalization method that can infer the normalization statistics of the target domain and use the inferred statistics to normalize test data during inference. We name our method Domain Conditioned Normalization (DCN). DCN simulates the inference process of the normalization statistics with a meta-learning framework during training, and uses the source statistics and one single target sample to infer the normalization statistics of the target domain with an optimization-free procedure during inference. Extensive experiments on three benchmarks demonstrate that our DCN can infer the reasonable target statistics so that it can achieve state-of-the-art performance for domain generalization. Furthermore, we conduct the ablation study to analyze the effects and characteristics of DCN.

Acknowledgement: This work is supported by the National Key R&D Program of China (No. 2019YFB1804304), 111 plan (No. BP0719010), and STCSM (No. 18DZ2270700, No. 21DZ1100100), and State Key Laboratory of UHD Video and Audio Production and Presentation.

References

1. Balaji, Y., Sankaranarayanan, S., Chellappa, R.: Metareg: Towards domain generalization using meta-regularization. Advances in Neural Information Processing Systems **31**, 998–1008 (2018) 3
2. Carlucci, F.M., D'Innocente, A., Bucci, S., Caputo, B., Tommasi, T.: Domain generalization by solving jigsaw puzzles. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 2229–2238 (2019) 3, 9, 11, 12
3. Chang, W.G., You, T., Seo, S., Kwak, S., Han, B.: Domain-specific batch normalization for unsupervised domain adaptation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 7354–7362 (2019) 5
4. Chen, Y., Wang, Y., Pan, Y., Yao, T., Tian, X., Mei, T.: A style and semantic memory mechanism for domain generalization. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 9164–9173 (2021) 3
5. Cubuk, E.D., Zoph, B., Shlens, J., Le, Q.V.: Randaugment: Practical automated data augmentation with a reduced search space. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops. pp. 702–703 (2020) 9
6. Du, Y., Zhen, X., Shao, L., Snoek, C.G.: Metanorm: Learning to normalize few-shot batches across domains. In: International Conference on Learning Representations (2020) 4, 10
7. Dubey, A., Ramanathan, V., Pentland, A., Mahajan, D.: Adaptive methods for real-world domain generalization. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 14340–14349 (2021) 2, 4
8. Fan, X., Wang, Q., Ke, J., Yang, F., Gong, B., Zhou, M.: Adversarially adaptive normalization for single domain generalization. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 8208–8217 (2021) 4, 9, 10
9. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016) 9
10. Hu, J., Shen, L., Sun, G.: Squeeze-and-excitation networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 7132–7141 (2018) 7
11. Huang, Z., Wang, H., Xing, E.P., Huang, D.: Self-challenging improves cross-domain generalization. In: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16. pp. 124–140. Springer (2020) 3, 10, 11, 12
12. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: International conference on machine learning. pp. 448–456. PMLR (2015) 2, 3, 5
13. Iwasawa, Y., Matsuo, Y.: Test-time classifier adjustment module for model-agnostic domain generalization. Advances in Neural Information Processing Systems **34** (2021) 2, 4
14. Jia, S., Chen, D.J., Chen, H.T.: Instance-level meta normalization. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4865–4873 (2019) 4
15. Khurana, A., Paul, S., Rai, P., Biswas, S., Aggarwal, G.: Sita: Single image test-time adaptation. arXiv preprint arXiv:2112.02355 (2021) 6

16. Kim, D., Yoo, Y., Park, S., Kim, J., Lee, J.: Selfreg: Self-supervised contrastive regularization for domain generalization. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 9619–9628 (2021) 3
17. Li, D., Yang, Y., Song, Y.Z., Hospedales, T.M.: Deeper, broader and artier domain generalization. In: Proceedings of the IEEE international conference on computer vision. pp. 5542–5550 (2017) 3, 9, 10
18. Li, D., Yang, Y., Song, Y.Z., Hospedales, T.M.: Learning to generalize: Meta-learning for domain generalization. In: Thirty-Second AAAI Conference on Artificial Intelligence (2018) 3
19. Li, H., Pan, S.J., Wang, S., Kot, A.C.: Domain generalization with adversarial feature learning (2018) 3, 12
20. Li, P., Li, D., Li, W., Gong, S., Fu, Y., Hospedales, T.M.: A simple feature augmentation for domain generalization. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 8886–8895 (2021) 3
21. Li, S., Xie, B., Lin, Q., Liu, C.H., Huang, G., Wang, G.: Generalized domain conditioned adaptation network. IEEE Transactions on Pattern Analysis and Machine Intelligence (2021) 7
22. Li, Y., Tian, X., Gong, M., Liu, Y., Liu, T., Zhang, K., Tao, D.: Deep domain generalization via conditional invariant adversarial networks. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 624–639 (2018) 3
23. Liang, S., Huang, Z., Liang, M., Yang, H.: Instance enhancement batch normalization: An adaptive regulator of batch noise. In: Proceedings of the AAAI Conference on Artificial Intelligence. pp. 4819–4827 (2020) 7
24. Matsuura, T., Harada, T.: Domain generalization using a mixture of multiple latent domains. In: Proceedings of the AAAI Conference on Artificial Intelligence. pp. 11749–11756 (2020) 11
25. Motiian, S., Piccirilli, M., Adjeroh, D.A., Doretto, G.: Unified deep supervised domain adaptation and generalization. In: Proceedings of the IEEE international conference on computer vision. pp. 5715–5725 (2017) 12
26. Nado, Z., Padhy, S., Sculley, D., D'Amour, A., Lakshminarayanan, B., Snoek, J.: Evaluating prediction-time batch normalization for robustness under covariate shift. arXiv preprint arXiv:2006.10963 (2020) 4
27. Nam, H., Kim, H.E.: Batch-instance normalization for adaptively style-invariant neural networks. arXiv preprint arXiv:1805.07925 (2018) 4, 10
28. Seo, S., Suh, Y., Kim, D., Kim, G., Han, J., Han, B.: Learning to optimize domain specific normalization for domain generalization. In: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXII 16. pp. 68–83. Springer (2020) 4, 10
29. Shahtalebi, S., Gagnon-Audet, J.C., Laleh, T., Faramarzi, M., Ahuja, K., Rish, I.: Sand-mask: An enhanced gradient masking strategy for the discovery of invariances in domain generalization. arXiv preprint arXiv:2106.02266 (2021) 3
30. Shankar, S., Piratla, V., Chakrabarti, S., Chaudhuri, S., Jyothi, P., Sarawagi, S.: Generalizing across domains via cross-gradient training. arXiv preprint arXiv:1804.10745 (2018) 12
31. Shi, Y., Seely, J., Torr, P.H., Siddharth, N., Hannun, A., Usunier, N., Synnaeve, G.: Gradient matching for domain generalization. arXiv preprint arXiv:2104.09937 (2021) 3
32. Sun, B., Saenko, K.: Deep coral: Correlation alignment for deep domain adaptation. In: European conference on computer vision. pp. 443–450. Springer (2016) 3

33. Sun, Y., Wang, X., Liu, Z., Miller, J., Efros, A., Hardt, M.: Test-time training with self-supervision for generalization under distribution shifts. In: International Conference on Machine Learning. pp. 9229–9248. PMLR (2020) 2, 4
34. Tang, Z., Gao, Y., Zhu, Y., Zhang, Z., Li, M., Metaxas, D.N.: Crossnorm and self-norm for generalization under distribution shifts. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 52–61 (2021) 7
35. Torralba, A., Efros, A.A.: Unbiased look at dataset bias. In: CVPR 2011. pp. 1521–1528. IEEE (2011) 3, 9, 11
36. Venkateswara, H., Eusebio, J., Chakraborty, S., Panchanathan, S.: Deep hashing network for unsupervised domain adaptation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 5018–5027 (2017) 3, 9, 12
37. Wang, D., Shelhamer, E., Liu, S., Olshausen, B., Darrell, T.: Tent: Fully test-time adaptation by entropy minimization. arXiv preprint arXiv:2006.10726 (2020) 2, 4
38. Wang, J., Lan, C., Liu, C., Ouyang, Y., Zeng, W., Qin, T.: Generalizing to unseen domains: A survey on domain generalization. arXiv preprint arXiv:2103.03097 (2021) 3
39. Wang, S., Yu, L., Li, C., Fu, C.W., Heng, P.A.: Learning from extrinsic and intrinsic supervisions for domain generalization. In: European Conference on Computer Vision. pp. 159–176. Springer (2020) 3, 10
40. Wang, X., Jin, Y., Long, M., Wang, J., Jordan, M.: Transferable normalization: Towards improving transferability of deep neural networks. Advances in Neural Information Processing Systems (2019) 7
41. Xu, Q., Zhang, R., Zhang, Y., Wang, Y., Tian, Q.: A fourier-based framework for domain generalization. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 14383–14392 (2021) 3, 9, 10, 12
42. You, F., Li, J., Zhao, Z.: Test-time batch statistics calibration for covariate shift. arXiv preprint arXiv:2110.04065 (2021) 6
43. Zhang, X., Cui, P., Xu, R., Zhou, L., He, Y., Shen, Z.: Deep stable learning for out-of-distribution generalization. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5372–5382 (2021) 11
44. Zhou, K., Liu, Z., Qiao, Y., Xiang, T., Loy, C.C.: Domain generalization: A survey. arXiv preprint arXiv:2103.02503 (2021) 3
45. Zhou, K., Yang, Y., Hospedales, T., Xiang, T.: Deep domain-adversarial image generation for domain generalisation. In: Proceedings of the AAAI Conference on Artificial Intelligence. pp. 13025–13032 (2020) 3, 12
46. Zhou, K., Yang, Y., Hospedales, T., Xiang, T.: Learning to generate novel domains for domain generalization. In: European Conference on Computer Vision. pp. 561–578. Springer (2020) 3, 12
47. Zhou, K., Yang, Y., Qiao, Y., Xiang, T.: Domain generalization with mixstyle. arXiv preprint arXiv:2104.02008 (2021) 3