Deep Domain-Adversarial Image Generation for Domain Generalisation

Kaiyang Zhou,[1] **Yongxin Yang,**[1] **Timothy Hospedales,**[2,3] **Tao Xiang**[1,3]
[1]University of Surrey, [2]University of Edinburgh, [3]Samsung AI Center, Cambridge
{k.zhou, yongxin.yang, t.xiang}@surrey.ac.uk, t.hospedales@ed.ac.uk

Abstract

Machine learning models typically suffer from the domain shift problem when trained on a source dataset and evaluated on a target dataset of different distribution. To overcome this problem, domain generalisation (DG) methods aim to leverage data from multiple source domains so that a trained model can generalise to unseen domains. In this paper, we propose a novel DG approach based on *Deep Domain-Adversarial Image Generation* (DDAIG). Specifically, DDAIG consists of three components, namely a label classifier, a domain classifier and a domain transformation network (DoTNet). The goal for DoTNet is to map the source training data to unseen domains. This is achieved by having a learning objective formulated to ensure that the generated data can be correctly classified by the label classifier while fooling the domain classifier. By augmenting the source training data with the generated unseen domain data, we can make the label classifier more robust to unknown domain changes. Extensive experiments on four DG datasets demonstrate the effectiveness of our approach.

Introduction

Most existing deep learning models assume that the training (source) and testing (target) data come from the same domain/dataset and thus follow the same distribution. However, in practice this assumption is often invalid. For example, a module for recognising pedestrians and traffic signs in an autonomous driving car may be deployed anywhere in the world under any weather condition. Considering each city and weather combination as a domain, it is impossible to collect training data of every domain for model training. Other domain changes can correspond to the change of image style/modality such as those shown in Figure 1 where a classifier trained on images of cartoon, photo and sketch is applied to art images. Unfortunately, it is well known that existing deep learning models are sensitive to domain changes/shifts (Li et al. 2017; Shankar et al. 2018; Balaji, Sankaranarayanan, and Chellappa 2018) in that they tend to overfit the source domains, resulting in poor generalisation.

A straightforward way to deal with the domain gap between source and target domains is to acquire labelled target data and perform supervised model fine-tuning. However, large-scale data collection and annotation for every new target domain is prohibitively expensive and time-consuming, which make the fine-tuning strategy infeasible. A more economical solution is to use unsupervised domain adaptation (UDA) methods (Long et al. 2015; Ganin and Lempitsky 2015; Hoffman et al. 2018; Gong et al. 2019; Chen et al. 2019), which only use unlabelled target data. Although the data annotation step is avoided, UDA still requires a data collection step followed by a model adaptation step for each new domain, which hinders its applicability.

As a result, domain generalisation (DG) (Muandet, Balduzzi, and Scholkopf 2013) has received an increasing interest lately. The goal of DG is to train a model using data from multiple source domains and deploy the model to an arbitrary unseen target domain without any adaption. Many existing DG methods adopt a core idea from the domain adaption (DA) research, which is to align source domain distributions at feature-level, assuming that a source domain invariant model can be learned (Li et al. 2018b; 2018c). However, without access to any target domain data, the model learned with domain alignment can still overfit the source domains. Alternatively, meta learning based methods have been recently employed to address DG where held-out source domains are used to simulate unseen target domains (Li et al. 2018a; Balaji, Sankaranarayanan, and Chel-

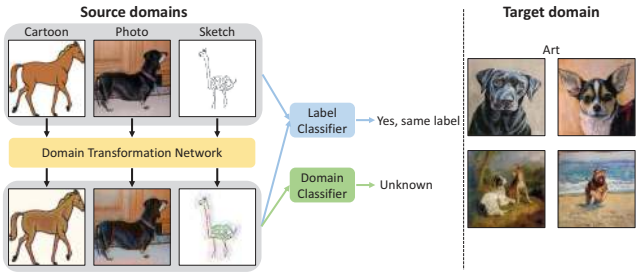

Figure 1: Given multiple source domains, e.g. Cartoon, Photo and Sketch, we learn a domain transformation network (DoTNet) to transform images to unseen domains, which maintain the class labels but change domain-related properties. Both original and transformed images are used to train a label classifier, which is applied to an unseen target domain, e.g. Art.

lappa 2018). However, meta learning models still focus on narrowing domain gaps among source domains and thus offer no guarantee for generalisation to unseen domains.

In this paper, we tackle the DG problem by *synthesising data from unseen domains*. We assume that augmenting the original training data of source domains with synthetic data from unseen domains could make the task model intrinsically more domain-generalisable (Tobin et al. 2017; Yue et al. 2019). To this end, a novel framework based on *Deep Domain-Adversarial Image Generation* (DDAIG) is introduced, which is illustrated in Figure 1. There are three components in DDAIG, which are label classifier, domain classifier and domain transformation network (DoTNet). Each component is a deep neural network. The label classifier and domain classifier are trained to predict the class labels and domain labels of the input data respectively. The functionality of DoTNet is to transform the input data in such a way that they can be recognised by the label classifier but fool the domain classifier. In particular, the transformation produced by DoTNet is designed to be perturbations with the same shape as the input. Therefore, the new data is generated by combining the perturbations with the original input. By doing so, we can efficiently generate additional training data that covers the (otherwise sparsely sampled) manifold of domains, which in turn allows a more domain-agnostic label classifier to be learned. We further show that DoTNet can be easily extended to incorporate other types of transformations, e.g., geometric transformations by adding spatial transformer network (STN) (Jaderberg et al. 2015). In practice, the three networks are trained jointly in an end-to-end manner. Unlike the domain alignment and meta learning methods, our DDAIG works directly at pixel-level, thus largely improving the interpretability of the model.

To evaluate DDAIG, we conduct extensive experiments on three DG benchmark datasets, namely PACS (Li et al. 2017), Office-Home (Venkateswara et al. 2017) and digit recognition among MNIST (LeCun et al. 1998), MNIST-M (Ganin and Lempitsky 2015), SVHN (Netzer et al. 2011) and SYN (Ganin and Lempitsky 2015). These datasets cover a variety of visual recognition tasks and contain different types of domain variation (see Figure 4). We demonstrate that DDAIG outperforms current state-of-the-art DG methods on all datasets. We also verify the effectiveness of DDAIG on a heterogeneous DG task, i.e., person re-identification where the source and target domains have different label spaces. Finally, we visualise the generated images and feature embeddings to provide insights on why our approach works. The code is available at https://github.com/KaiyangZhou/DG-research-pytorch.

Related Work

Early kernel alignment (Muandet, Balduzzi, and Scholkopf 2013; Gan, Yang, and Gong 2016) and examplar SVM based (Xu et al. 2014; Niu, Li, and Xu 2015) DG models have been followed mainly by deep neural network based ones. The current deep DG studies can be generally divided into three groups: (i) domain alignment, (ii) meta learning and (iii) data augmentation.

Domain alignment has been extensively studied for domain adaptation (DA) problems, where some labelled or unlabelled data are accessible during training. These DA methods aim to either (i) minimise the distance (e.g., maximum mean discrepancy (MMD) (Pan, Kwok, and Yang 2008)) between source and target distributions, or (ii) fool a domain classifier that tries to discriminate different domains (Ganin and Lempitsky 2015). As shown in (Motiian et al. 2017), domain alignment based DA models can be easily modified for DG by iteratively training on every pair of source domains. Among the recent alignment based deep DG models, (Li et al. 2018b) proposed to minimise MMD of all possible pairs within source domains, meanwhile an adversarial autoencoder was used to ensure that the learned features follow the Laplace distribution. (Li et al. 2018c) considered aligning the conditional distributions as well as the marginal ones via adversarial training. Though domain alignment is a sensible strategy for DA, the potential risk of applying it to DG is that the model might overfit all seen domains yet still generalise poorly to the unseen domains.

Meta learning in computer vision has been widely exploited for few-shot learning (Finn, Abbeel, and Levine 2017). Recently, meta learning has been adapted to address the DG setting (Li et al. 2018a; Balaji, Sankaranarayanan, and Chellappa 2018). Since the final objective of a DG model is to generalise to unseen domains, the key idea of using meta learning is to simulate domain shift during training to prepare models for domain shift during testing. Specifically, source domains are separated into two disjoint sets, namely meta-train and meta-validation, and a model is optimised on meta-train so as to boost the performance on meta-validation. One early work in this direction is MLDG (Li et al. 2018a), which is based on MAML (Finn, Abbeel, and Levine 2017). Recently, MetaReg (Balaji, Sankaranarayanan, and Chellappa 2018) proposed to learn a customised regulariser to improve DG. A meta learning based DG approach is appealing as it reduces the efforts in manual design. However, as a black-box approach it is hard to diagnose exactly how it improves the DG performance. Importantly, using only the original source domain data, it still has the risk of overfitting source domains.

Data augmentation is a common practice to train deep neural networks, e.g. flipping and rotation. However, conventional data augmentation methods only deal with simple geometric changes within the same dataset (Volpi and Murino 2019). When the domain gap is large such as those illustrated in Figure 4 containing image style variations, learning-based augmentation strategies are required. Very recently, inspired by adversarial attacks (Goodfellow, Shlens, and Szegedy 2015), (Shankar et al. 2018) introduced CrossGrad to generate a new sample $\tilde{x}$ by adding to the original sample its gradient from a domain classifier $h(\cdot)$, i.e., $\tilde{x} \leftarrow x + \epsilon \frac{\partial h(x)}{\partial x}$. A similar approach with an additional regularisation term in $h(\cdot)$ was proposed in (Volpi et al. 2018) for the single source domain case. The main drawback of these methods is their direct and simple dependence on the gradient, which only makes simple perturbations that cannot account for semantic changes like style or font shift (see Fig-

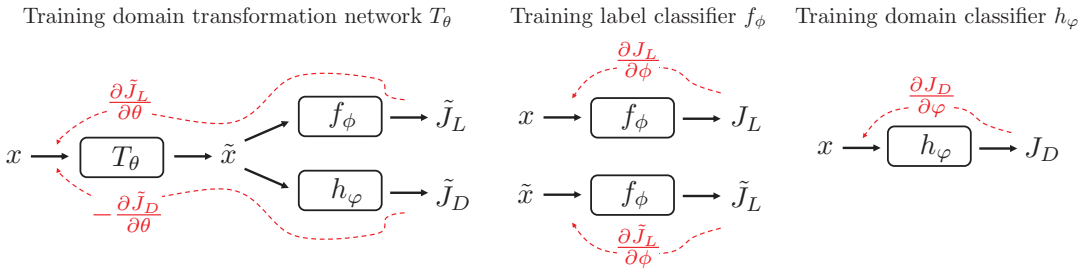

Figure 2: Overview of our framework. A domain transformation network T_θ is trained by minimising the label classification loss $\tilde{J}_L$ while maximising the domain classification loss $\tilde{J}_D$ on the transformed data $\tilde{x}$. The label classifier f_ϕ is learned by minimising the label classification loss given both original and transformed data. The domain classifier h_φ is trained to classify each instance into one of source domains. The red dashed arrows represent the gradient flow.

ure 9). Moreover, being based on adversarial attack models that are deliberately designed to make imperceptible modifications to an image, the perturbations are too subtle to be representative of real-world domain shift. In contrast, by *learning* a full CNN model (i.e. DoTNet) to generate the 'shift', we can produce more sophisticated and more overt perturbations to synthesise new data and the results are easier to interpret. We demonstrate clear advantages of our approach over CrossGrad both quantitatively and qualitatively. We also show that our transformation CNN can be easily extended to incorporate geometric transformations such as rotation (see Figure 5), which is impossible with gradient-based perturbation methods.

Methodology

Our idea to tackle domain generalisation (DG) is based on *Deep Domain-Adversarial Image Generation* (DDAIG), which aims to train a domain transformation network (DoTNet) to synthesise data from unseen domains given some input and use both original and synthetic data to learn a domain-invariant classifier. To learn DoTNet, we simultaneously train a label classifier and a domain classifier, which are tasked to recognise the class labels and domains of the input data, respectively. The learning objective for DoTNet is to transform the input data in such a way that the synthetic data keeps the same labels as the input but fools the domain classifier. As the synthetic data has labels, it can be combined with the original data to train the label classifier using supervised learning. As a result, the label classifier can learn representations that are more invariant to domain shift than that trained with the original data only[1]. An overview of our DDAIG framework is illustrated in Figure 2. The learning procedure for each component is detailed below.

Domain transformation network. Let T_θ be the DoTNet parameterised by θ, f_ϕ the label classifier parameterised by ϕ, h_φ the domain classifier parameterised by φ, y the class label of input x and d the domain label, the objective function for T_θ is

$$\min_\theta \ \tilde{J}_L(f_\phi(T_\theta(x)), y) - \tilde{J}_D(h_\varphi(T_\theta(x)), d), \qquad (1)$$

where $\tilde{J}_L$ and $\tilde{J}_D$ are cross-entropy losses for label and domain classification, respectively. We use differentiable neural networks to construct T_θ, f_ϕ and h_φ, thus the gradients can be back-propagated through f_ϕ and h_φ and all the way to T_θ. The specific architecture design of T_θ will be discussed later.

Label classifier. The label classifier f_ϕ is fed with both original and synthetic data. The loss function is

$$\min_\phi \ (1-\alpha) J_L(f_\phi(x), y) + \alpha \tilde{J}_L(f_\phi(T_\theta(x)), y), \qquad (2)$$

where α is a balance weight, which is fixed to 0.5.

Domain classifier. The domain classifier h_φ is required to capture domain-discriminative features, thus its learning objective is to minimise the domain classification loss w.r.t φ,

$$\min_\varphi \ J_D(h_\varphi(x), d). \qquad (3)$$

Note that our domain classifier is analogous to the discriminator in the classic GAN framework (Goodfellow et al. 2014) but differs in that we do multi-class classification (Odena, Olah, and Shlens 2017) (on source domains[2]) while GAN's discriminator performs binary classification (real or fake). This difference ensures that maximising the domain classification loss does not simply force the synthetic data to fall into another single domain distribution. Suppose there are three source domains, given a synthetic instance from the first domain we maximise $-\log \frac{e^{z_1}}{e^{z_1}+e^{z_2}+e^{z_3}}$ (z_i denotes logit), which essentially minimises z_1 whilst giving *equal gradients* to maximise z_2 and z_3 simultaneously. As such, neither one of the gradients to z_2 and z_3 is dominant.

Architecture design. For the label classifier, any network architecture suitable for the given recognition problem can be adopted. Throughout this paper, the domain and label classifiers share the same architecture. To construct DoTNet, we use fully convolutional network (FCN) (Long, Shelhamer, and Darrell 2015). Instead of directly generating data, which is often difficult because the data can be high-dimensional such as RGB image, we use FCN to generate perturbations, which are added to the input, resulting in

$$\tilde{x} = x + \lambda T_\theta(x), \qquad (4)$$

[1]To clarify, a classifier means the combination of a feature extraction backbone and a softmax classification layer, unless specified otherwise.

[2]In DG tasks, we often assume that we have access to multiple source domains.

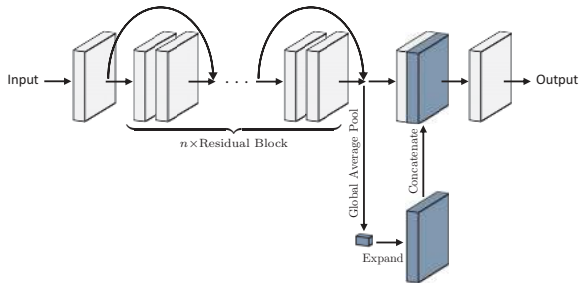

Figure 3: Architecture of domain transformation network.

where λ is a positive weight typically set between 0.1 and 0.7. This is inspired by the residual feature learning (He et al. 2016) but the residual connection links the input directly to the output. This design is also related to adversarial attack methods (Szegedy et al. 2014; Goodfellow, Shlens, and Szegedy 2015). However, different from adversarial perturbations, which are usually imperceptible, our DoTNet is allowed to produce visually perceptible perturbations, which can better represent the real-world domain shift (see Figure 8). Note that (4) will replace the $T_\theta(x)$ in (1) and (2).

More concretely, as shown in Figure 3, the architecture of DoTNet starts with a 3×3 conv layer, followed by n 2-conv residual blocks (He et al. 2016) to extract mid-level features. All residual blocks use 3×3 kernels. The output is branched into an identity layer and a global average-pooling layer. The latter produces a context vector encapsulating the global information (Liu, Rabinovich, and Berg 2016). The context vector is expanded spatially and then concatenated back to the main stream, which is further processed by a 1×1 conv layer for feature fusion. Finally, a 1×1 conv layer is used to generate the perturbation. All layers use ReLU as the non-linearity function except the last one which uses the Tanh function. Similar to (Zhu et al. 2017), we insert instance normalisation layer (Ulyanov, Vedaldi, and Lempitsky 2017) after every conv layer excluding the last one. The stride is set to 1 for all layers. All 3×3 kernels use the reflection padding of size 1.

In this paper, we mainly investigate this perturbation design for T. However, our framework is generic and does not impose any restriction on the architecture (as long as it is differentiable). The design of T thus mostly depends on the specific tasks at hand. For instance, T can take the form of STN (Jaderberg et al. 2015) to deal with geometric transformations; or even a combination of STN and the perturbation architecture in Figure 3.

The full algorithm of DDAIG is presented in Algorithm 1. Note that the warm-up scheme mainly aims to make the data generated by DoTNet more reliable before being fed to the classifier.

Algorithm 1 Deep Domain-Adversarial Image Generation

Input: source domains $\mathcal{S}$, label classifier f_ϕ, domain classifier h_φ, DoTNet T_θ, learning rate η, hyperparameter λ, loss balance weight α, maximum iteration K, warmup iteration K_m.
Output: label classifier f_ϕ.
for $k = 1$ **to** K **do**
  $(x, y, d) \sim \mathcal{S}$ *// Randomly sample a minibatch*
  $\tilde{x} = x + \lambda T_\theta(x)$ *// Transform the minibatch*
  $\theta = \theta - \eta \nabla_\theta (\tilde{J}_L - \tilde{J}_D)$ *// Update DoTNet*
  if $k < K_\text{m}$ **then**
    $\phi = \phi - \eta \nabla_\phi J_L$ *// Update label classifier*
  else
    $\tilde{x} = x + \lambda T_\theta(x)$ *// Transform the minibatch using updated DoTNet*
    $\phi = \phi - \eta \nabla_\phi ((1-\alpha) J_L + \alpha \tilde{J}_L)$ *// Update label classifier using both original and synthetic data*
  end if
  $\varphi = \varphi - \eta \nabla_\varphi J_D$ *// Update domain classifier*
end for

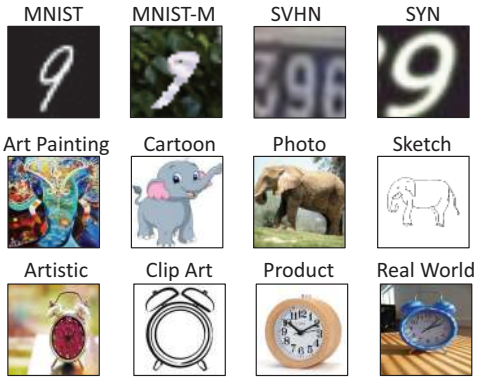

Figure 4: Example images from Digits-DG (1st row), PACS (2nd row) and Office-Home (3rd row) show that large domain gaps exist, challenging domain generalisation.

Experiments

Datasets and Settings

We first evaluate our approach DDAIG on three conventional DG benchmark datasets, which cover a variety of recognition problems. (1) We conduct leave-one-domain-out digit recognition on *MNIST* (LeCun et al. 1998), *MNIST-M* (Ganin and Lempitsky 2015), *SVHN* (Netzer et al. 2011) and *SYN* (Ganin and Lempitsky 2015), which differ drastically in font style and background (see Figure 4 1st row). MNIST contains handwritten digit images. MNIST-M is a variant of MNIST by blending the images with random colour patches. SVHN contains street view house number images. SYN consists of synthetic digit images with varying fonts, backgrounds and stroke colours. This benchmark is called **Digits-DG** hereafter. We randomly select 600 images per class from each dataset and split the data into 80% for training and 20% for validation. All models are trained on the training data of three domains and evaluated on all images of the remaining domain. (2) **PACS** (Li et al. 2017) consists of four domains, namely *Photo* (1,670 images), *Art Painting* (2,048 images), *Cartoon* (2,344 images) and *Sketch* (3,929 images). Each domain contains seven categories. Following the prior work (Li et al. 2017; Carlucci et al. 2019; Li et al. 2019a), we choose one domain as test domain and use the remaining three domains as source domains. To fairly compare with published methods, our models are trained using data only from the training split. The domain

Table 1: Leave-one-domain-out results on Digits-DG dataset (with 95% confidence intervals).

Method	MNIST	MNIST-M	SVHN	SYN	Avg.
Vanilla	95.8±.3	58.8±.5	61.7±.5	78.6±.6	73.7
CCSA	95.2±.2	58.2±.6	65.5±.2	79.1±.8	74.5
MMD-AAE	96.5±.1	58.4±.1	65.0±.1	78.4±.2	74.6
CrossGrad	**96.7**±.1	61.1±.5	65.3±.5	80.2±.2	75.8
DDAIG (*ours)*	96.6±.2	**64.1**±.4	**68.6**±.6	**81.0**±.5	**77.6**

shift mainly corresponds to image style changes as depicted in Figure 4 2nd row. (3) **Office-Home** (Venkateswara et al. 2017), originally introduced for domain adaptation, is getting popular in the DG community (D'Innocente and Caputo 2018; Carlucci et al. 2019). It contains four domains, which are *Artistic*, *Clipart*, *Product* and *Real World*. Each domain has 65 classes, which are related to office and home objects. There are around 15,500 images in total. The domain variations mainly take place in background, viewpoint and image style. See Figure 4 (the third row) for example images.

For performance measure, we report top-1 classification accuracy (%) averaged over five runs and 95% confidence intervals. We compare our DDAIG with state-of-the-art DG methods with reported results on these datasets or codes. These methods include CCSA (Motiian et al. 2017), MMD-AAE (Li et al. 2018b), CrossGrad (Shankar et al. 2018), MetaReg (Balaji, Sankaranarayanan, and Chellappa 2018), D-SAM (D'Innocente and Caputo 2018), JiGen (Carlucci et al. 2019) and Epi-FCR (Li et al. 2019a). We also include a strong baseline called Vanilla, which directly combines data from all source domains for model training without any DG-targeting tricks.

Evaluation on Digits-DG

Implementation. Images are resized to 32×32 and converted to RGB by replicating channels. The classification network is constructed by four 3×3 conv layers (64 kernels), each followed by ReLU and 2×2 max-pooling. A softmax classification layer is attached on the top, which takes the flattened vector as input. The networks are trained from scratch using SGD, initial learning rate of 0.05, batch size of 128 and weight decay of 5e-4 for 50 epochs. The learning rate is decayed by 0.1 every 20 epochs.

Results. Table 1 shows that our DDAIG achieves the best overall performance (Avg.), outperforming the second best CrossGrad by a clear margin of 1.8% and all domain alignment methods (CCSA & MMD-AAE) by more than 3%. On the most difficult target domains, namely MNIST-M and SVHN which contain complex backgrounds and cluttered digits respectively, DDAIG obtains large margins over the competitors, notably with +5.3% and +6.9% improvements compared with the Vanilla model. This demonstrates the effectiveness of the generated unseen domain data, which essentially increases the diversity of source domains.

Evaluation on PACS

Implementation. Images are resized to 224×224. Following (Carlucci et al. 2019; Li et al. 2019a), we use the ImageNet-pretrained ResNet18 (He et al. 2016) as the classification network. The networks are trained with SGD, initial learning rate of 5e-4, batch size of 16 and weight decay of 5e-4 for 25 epochs. The learning rate is decayed by 0.1 at the 20th epoch. For data augmentation, we use random crop on images rescaled by a factor of 1.25 and random horizontal flip. During the first three epochs, the label classifier is only fed with real data.

Table 2: Leave-one-domain-out results on PACS dataset (with 95% confidence intervals). †: results are reported in their papers. ‡: use train+val for training.

Method	Art	Cartoon	Photo	Sketch	Avg.
MetaReg†‡	83.7±.1	77.2±.3	95.5±.2	70.3±.3	81.7
Vanilla	77.0±.6	75.9±.6	**96.0**±.1	69.2±.6	79.5
CCSA	80.5±.6	76.9±.6	93.6±.4	66.8±.9	79.4
MMD-AAE	75.2±.3	72.7±.3	**96.0**±.1	64.2±.2	77.0
CrossGrad	79.8±.8	76.8±.8	**96.0**±.2	70.2±.4	80.7
D-SAM†	77.3	72.4	95.3	**77.8**	80.7
JiGen†	79.4	75.3	**96.0**	71.6	80.5
Epi-FCR†	82.1	77.0	93.9	73.0	81.5
DDAIG (*ours*))	**84.2**±.3	**78.1**±.6	95.3±.4	74.7±.8	**83.1**

Table 3: Leave-one-domain-out results on Office-Home dataset (with 95% confidence intervals). †: results are reported in their papers.

Method	Artistic	Clipart	Product	Real World	Avg.
Vanilla	58.9±.3	49.4±.1	74.3±.1	**76.2**±.2	64.7
CCSA	**59.9**±.3	49.9±.4	74.1±.2	75.7±.2	64.9
MMD-AAE	56.5±.4	47.3±.3	72.1±.3	74.8±.2	62.7
CrossGrad	58.4±.7	49.4±.4	73.9±.2	75.8±.1	64.4
D-SAM†	58.0	44.4	69.2	71.5	60.8
JiGen†	53.0	47.5	71.5	72.8	61.2
DDAIG (*ours*))	59.2±.1	**52.3**±.3	**74.6**±.3	76.0±.1	**65.5**

Results. We summarise our findings from Table 2 as follows. (1) Our DDAIG is clearly the best method, beating the second best methods MetaReg and Epi-FCR by around 1.5% (on Avg.). It is noted that MetaReg benefits from additional training data from the validation split. The recently proposed Epi-FCR uses episodic training to improve the Vanilla model. DDAIG outperforms Epi-FCR on all domains by a clear margin, suggesting that data augmentation with unseen domain data is much more effective. (2) Again, DDAIG achieves large improvements (3.7%+ on Avg.) against all domain alignment methods. (3) Compared with CrossGrad, DDAIG yields large improvements on all domains except Photo. In particular, the margins are 4.4% on Art, 1.3% on Cartoon and 4.5% on Sketch. This justifies that learning a dedicated CNN for perturbation generation is much more useful than gradient-based perturbations.

Evaluation on Office-Home

Implementation. Following (D'Innocente and Caputo 2018; Carlucci et al. 2019), we randomly split the data into 90% for training and 10% for validation. The commonly used leave-one-domain-out protocol is adopted for evaluation. For fair comparison with published methods, we only use the training split of source domains for model training. Other implementation details for network training are the same as those for PACS.

Results. From Table 3, we observe that the Vanilla model

Table 4: Results on cross-domain person re-ID datasets.

Method	Market1501→Duke				Duke→Market1501			
	R1	R5	R10	mAP	R1	R5	R10	mAP
Vanilla	48.5	62.3	67.4	26.7	57.7	73.7	80.0	26.1
CrossGrad	48.5	63.5	69.5	27.1	56.7	73.5	79.5	26.3
DDAIG (*ours*)	**50.6**	**65.2**	**70.3**	**28.6**	**60.9**	**77.1**	**83.2**	**29.0**

Table 5: Results on Digits-DG using different λ's.

Method	MNIST	MNIST-M	SVHN	SYN	Avg.
Baseline	95.8	58.8	61.7	78.6	73.7
DDAIG $\lambda = 0.1$	96.3	62.3	68.6	79.8	76.8
DDAIG $\lambda = 0.3$	96.4	61.9	68.0	81.0	76.8
DDAIG $\lambda = 0.5$	96.6	61.2	68.0	80.5	76.6
DDAIG $\lambda = 0.7$	96.4	64.1	65.9	80.8	76.8

achieves very strong performance, largely outperforming most DG methods including MMD-AAE, D-SAM and JiGen. This makes sense because the domain gap is much smaller compared to that in PACS, especially among Artistic, Product and Real World, where the variations mainly take place in background and viewpoint. Among all methods, only DDAIG achieves a clear margin against Vanilla. In particular, it is worth noting that DDAIG achieves huge improvement (+2.9%) on Clipart, which contains the largest domain gap as opposed to the source domains (see Figure 4). Therefore, the results strongly demonstrate the versatility of our DDAIG framework. Compared with CrossGrad, DDAIG achieves better performance on all domains.

Evaluation on Heterogeneous DG

In this section, we evaluate our approach on the cross-dataset person re-identification (re-ID) task, which is essentially a heterogeneous DG problem due to disjoint label spaces between training and test identities (Li et al. 2019b). Person re-ID aims to match people across non-overlapping camera views. In this task, we treat each camera view as a domain.

Datasets. We use two commonly used re-ID datasets, namely *Market1501* (Zheng et al. 2015) and *DukeMTMC-reID* (Duke) (Ristani et al. 2016; Zheng, Zheng, and Yang 2017). Market1501 contains 32,668 images of 1,501 identities, which are captured by 6 cameras. Duke contains 36,411 images of 1,812 identities, which are captured by 8 cameras. Each dataset is split into training set, query set and gallery set based on the standard protocols (Zheng et al. 2015; Zheng, Zheng, and Yang 2017). For evaluation, we train models using one dataset and perform test on the other. Cumulative Matching Characteristics (CMC) ranks and mean Average Precision (mAP) are used as the performance measure.

Implementation. Images are resized to 256×128. We adopt the state-of-the-art re-ID model OSNet (Zhou et al. 2019b; 2019a) as the CNN backbone. Following (Zhou et al. 2019b; 2019a), we train the re-ID model using the standard classification pipeline where each person identity is regarded as a class. Therefore, the entire training algorithm remains the same as before. At test time, the features extracted from the re-ID model are used to compute Euclidean distance for image matching. The code is based on Torchreid (Zhou and Xiang 2019).

Table 6: Accuracy on rotated MNIST when MNIST-M, SVHN and SYN are used as the source domains.

Method	0°	20°	30°	40°	50°
CrossGrad	96.7	82.7	65.1	45.6	30.1
DDAIG *w/o STN*	96.6	87.7	72.1	54.1	40.3
DDAIG *w/ STN*	96.4	87.7	76.7	61.1	46.6

Figure 5: Transformations produced by an extended DoTNet. STN: Spatial Transformer Network. P: Perturbation.

Results. We compare our method with CrossGrad and the strong vanilla model. The overall results are shown in Table 4. It is clear that only our DDAIG consistently improves upon the vanilla baseline on both settings, with noticeable margins. It is widely acknowledged that cross-domain re-ID is a challenging problem (Zhong et al. 2019; Zhou et al. 2019a). Without using target data, it is difficult to gain improvement over the vanilla model. Therefore, the results strongly demonstrate the versatility of our DDAIG: It is not only effective for the conventional DG tasks but also useful to heterogeneous DG problems such as person re-ID.

Further Analysis

Impact of λ. Table 5 shows the results of varying λ in Eq. 4 from 0.1 to 0.7. When the target domain is less dissimilar to the source domains such as MNIST and SYN, the result does not vary too much with different λs. However, when the target domain has a larger domain gap than the sources, e.g. MNIST-M and SVHN, our model shows a moderate sensitivity to λ. It is important to note that all results are better than the Vanilla baseline.

Dealing with geometric transformations. Image perturbation is less useful in simulating geometric transformations such as rotation. To overcome this limitation, we extend the perturbation CNN (Figure 3) by inserting STN (Jaderberg et al. 2015) before it. Therefore, the new DoTNet first geometrically transforms the input and then adds perturbation. Note that such a transformation is impossible with gradient-based perturbation methods (Shankar et al. 2018). We test this design on Digits-DG where the target domain is rotated MNIST and the source domains are MNIST-M, SVHN and SYN. Note that the 'rotation' shift is never observed among the sources. The results are shown in Table 6. First of all, we observe that all methods' performance drops as the rotation degree increases. This is expected because increasing the rotation degree essentially enlarges the domain gap with the source data, making the target domain more challenging. Comparing DDAIG (without STN) with CrossGrad, the performance drop of the latter is much larger. This indicates that the CNN-learned perturbation does not contain solely the style changes but also sophisticated transformations, which make the task model more robust to the geometric domain shift. With STN, the performance drop is significantly reduced (especially on $40°$ and $50°$) which demonstrates the

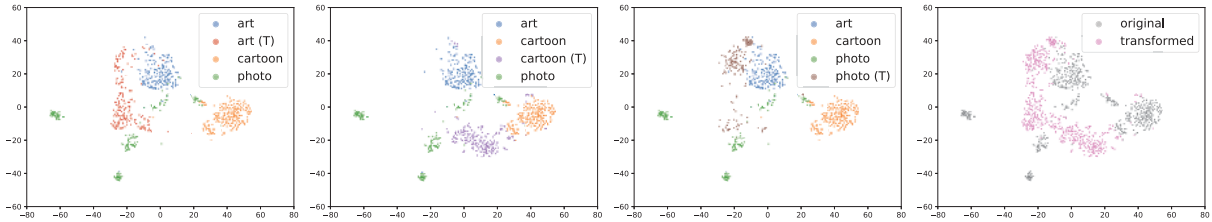

Figure 6: T-SNE visualisation in the domain space using PACS's validation set. The first three images compare transformed data (T) with original data for each source domain. The last image shows an overall comparison between original (grey) and transformed (pink) data.

Table 7: Comparison between models trained using source data, novel data and source+novel data.

Souce	Novel	MNIST	MNIST-M	SVHN	SYN	Avg.
✓		95.8	58.8	61.7	78.6	73.7
	✓	95.6	58.3	57.9	79.9	72.9
✓	✓	96.6	64.1	68.6	81.0	77.6

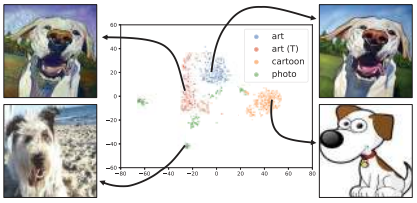

Figure 7: Transformed image vs. original source images.

flexibility of the DDAIG framework. See Figure 5 for the visualisation of the transformations.

Importance of combining source and novel data. Table 7 shows that training with the novel data only does not bring any gain at all. This is expected because the performance gain of DDAIG mainly comes from the aggregation of source domains and the generated novel domains.

Visualisation. To better understand why DDAIG works for DG, we visualise the feature embeddings in the domain space using t-SNE (Maaten and Hinton 2008) (see Figure 6). It is clear that the new data distributions do not overlap with any of the existing source domains. Instead, they are distributed over the unfilled domain space, indicating exploration of unseen domains. Consequently, the generated unseen-domain data along with the source-domain data allows the model to learn more domain-generalisable representations, which explains why DDAIG achieves excellent performance on all DG benchmark datasets. Figure 7 shows four example images from the first domain space where the new "art" image clearly differs from the other source images in terms of image style.

Figure 8 provides a clearer view of how images are transformed by the perturbations. Comparing the transformed images with the original images, we observe that the domain-related information has been drastically changed while the category-specific properties are well maintained, which is consistent with the motivation of our method. The perturbations are instance-specific and can represent complex transformations such as colour and texture. For instance, the perturbation for the elephant image tends to add green and pink colours to the background and enhance the textures on the elephant body.

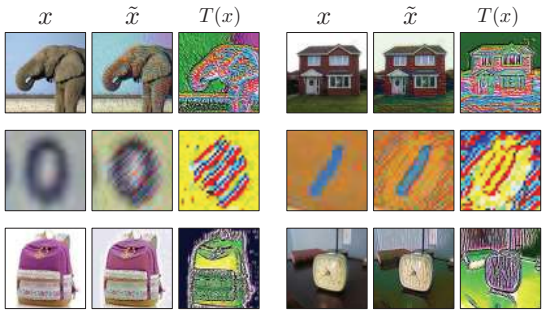

Figure 8: Examples of transformed images from PACS (1st row), Digits-DG (2nd row) and Office-Home (3rd row). x, $\tilde{x}$ and $T(x)$ denote original image, transformed image and transformation (perturbation), respectively.

Figure 9: Comparison between perturbations generated by CNN (ours) and adversarial perturbations.

Comparison with adversarial perturbations. Figure 9 compares the perturbations between DDAIG and CrossGrad. It is obvious that CrossGrad's perturbation does not contain meaningful patterns and look like salt-and-pepper noise, resembling those of adversarial attack methods (Goodfellow et al. 2014). In contrast, our perturbation is instance-specific and has obvious effects on the transformed image, which is more representative of the real-world domain shift.

Conclusion

We presented DDAIG, a novel DG method to synthesise data from unseen domains for data augmentation. Unlike current data augmentation-based DG methods, DDAIG learns a full transformation CNN to model the domain shift. Extensive experiments on three DG datasets showed that our method can improve the generalisation of CNN models on unseen domains, outperforming current state-of-the-art DG methods. Results on the cross-domain person re-ID task further demonstrated the versatility of DDAIG beyond DG.

References

Balaji, Y.; Sankaranarayanan, S.; and Chellappa, R. 2018. Metareg: Towards domain generalization using meta-regularization. In *NeurIPS*.

Carlucci, F. M.; D'Innocente, A.; Bucci, S.; Caputo, B.; and Tommasi, T. 2019. Domain generalization by solving jigsaw puzzles. In *CVPR*.

Chen, X.; Wang, S.; Long, M.; and Wang, J. 2019. Transferability vs. discriminability: Batch spectral penalization for adversarial domain adaptation. In *ICML*.

D'Innocente, A., and Caputo, B. 2018. Domain generalization with domain-specific aggregation modules. In *GCPR*.

Finn, C.; Abbeel, P.; and Levine, S. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *ICML*.

Gan, C.; Yang, T.; and Gong, B. 2016. Learning attributes equals multi-source domain generalization. In *CVPR*.

Ganin, Y., and Lempitsky, V. S. 2015. Unsupervised domain adaptation by backpropagation. In *ICML*.

Gong, R.; Li, W.; Chen, Y.; and Gool, L. V. 2019. Dlow: Domain flow for adaptation and generalization. In *CVPR*.

Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; and Bengio, Y. 2014. Generative adversarial nets. In *NeurIPS*.

Goodfellow, I. J.; Shlens, J.; and Szegedy, C. 2015. Explaining and harnessing adversarial examples. In *ICLR*.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *CVPR*.

Hoffman, J.; Tzeng, E.; Park, T.; Zhu, J.-Y.; Isola, P.; Saenko, K.; Efros, A.; and Darrell, T. 2018. Cycada: Cycle-consistent adversarial domain adaptation. In *ICML*.

Jaderberg, M.; Simonyan, K.; Zisserman, A.; et al. 2015. Spatial transformer networks. In *NeurIPS*.

LeCun, Y.; Bottou, L.; Bengio, Y.; and Haffner, P. 1998. Gradient-based learning applied to document recognition. In *IEEE*.

Li, D.; Yang, Y.; Song, Y.-Z.; and Hospedales, T. M. 2017. Deeper, broader and artier domain generalization. In *ICCV*.

Li, D.; Yang, Y.; Song, Y.-Z.; and Hospedales, T. M. 2018a. Learning to generalize: Meta-learning for domain generalization. In *AAAI*.

Li, H.; Jialin Pan, S.; Wang, S.; and Kot, A. C. 2018b. Domain generalization with adversarial feature learning. In *CVPR*.

Li, Y.; Tiana, X.; Gong, M.; Liu, Y.; Liu, T.; Zhang, K.; and Tao, D. 2018c. Deep domain generalization via conditional invariant adversarial networks. In *ECCV*.

Li, D.; Zhang, J.; Yang, Y.; Liu, C.; Song, Y.-Z.; and Hospedales, T. M. 2019a. Episodic training for domain generalization. In *ICCV*.

Li, Y.; Yang, Y.; Zhou, W.; and Hospedales, T. 2019b. Feature-critic networks for heterogeneous domain generalization. In *ICML*.

Liu, W.; Rabinovich, A.; and Berg, A. C. 2016. Parsenet: Looking wider to see better. In *ICLR-W*.

Long, M.; Cao, Y.; Wang, J.; and Jordan, M. 2015. Learning transferable features with deep adaptation networks. In *ICML*.

Long, J.; Shelhamer, E.; and Darrell, T. 2015. Fully convolutional networks for semantic segmentation. In *CVPR*.

Maaten, L. v. d., and Hinton, G. 2008. Visualizing data using t-sne. *JMLR*.

Motiian, S.; Piccirilli, M.; Adjeroh, D. A.; and Doretto, G. 2017. Unified deep supervised domain adaptation and generalization. In *ICCV*.

Muandet, K.; Balduzzi, D.; and Scholkopf, B. 2013. Domain generalization via invariant feature representation. In *ICML*.

Netzer, Y.; Wang, T.; Coates, A.; Bissacco, A.; Wu, B.; and Ng, A. Y. 2011. Reading digits in natural images with unsupervised feature learning. In *NeurIPS-W*.

Niu, L.; Li, W.; and Xu, D. 2015. Multi-view domain generalization for visual recognition. In *ICCV*.

Odena, A.; Olah, C.; and Shlens, J. 2017. Conditional image synthesis with auxiliary classifier gans. In *ICML*.

Pan, S. J.; Kwok, J. T.; and Yang, Q. 2008. Transfer learning via dimensionality reduction. In *AAAI*.

Ristani, E.; Solera, F.; Zou, R. S.; Cucchiara, R.; and Tomasi, C. 2016. Performance measures and a data set for multi-target, multi-camera tracking. In *ECCV-W*.

Shankar, S.; Piratla, V.; Chakrabarti, S.; Chaudhuri, S.; Jyothi, P.; and Sarawagi, S. 2018. Generalizing across domains via cross-gradient training. In *ICLR*.

Szegedy, C.; Zaremba, W.; Sutskever, I.; Bruna, J.; Erhan, D.; Goodfellow, I. J.; and Fergus, R. 2014. Intriguing properties of neural networks. In *ICLR*.

Tobin, J.; Fong, R.; Ray, A.; Schneider, J.; Zaremba, W.; and Abbeel, P. 2017. Domain randomization for transferring deep neural networks from simulation to the real world. In *IROS*.

Ulyanov, D.; Vedaldi, A.; and Lempitsky, V. 2017. Improved texture networks: Maximizing quality and diversity in feed-forward stylization and texture synthesis. In *CVPR*.

Venkateswara, H.; Eusebio, J.; Chakraborty, S.; and Panchanathan, S. 2017. Deep hashing network for unsupervised domain adaptation. In *CVPR*.

Volpi, R., and Murino, V. 2019. Model vulnerability to distributional shifts over image transformation sets. In *ICCV*.

Volpi, R.; Namkoong, H.; Sener, O.; Duchi, J.; Murino, V.; and Savarese, S. 2018. Generalizing to unseen domains via adversarial data augmentation. In *NeurIPS*.

Xu, Z.; Li, W.; Niu, L.; and Xu, D. 2014. Exploiting low-rank structure from latent domains for domain generalization. In *ECCV*.

Yue, X.; Zhang, Y.; Zhao, S.; Sangiovanni-Vincentelli, A.; Keutzer, K.; and Gong, B. 2019. Domain randomization and pyramid consistency: Simulation-to-real generalization without accessing target domain data. In *ICCV*.

Zheng, L.; Shen, L.; Tian, L.; Wang, S.; Wang, J.; and Tian, Q. 2015. Scalable person re-identification: A benchmark. In *ICCV*.

Zheng, Z.; Zheng, L.; and Yang, Y. 2017. Unlabeled samples generated by gan improve the person re-identification baseline in vitro. In *ICCV*.

Zhong, Z.; Zheng, L.; Luo, Z.; Li, S.; and Yang, Y. 2019. Invariance matters: Exemplar memory for domain adaptive person re-identification. In *CVPR*.

Zhou, K., and Xiang, T. 2019. Torchreid: A library for deep learning person re-identification in pytorch. *arXiv preprint arXiv:1910.10093*.

Zhou, K.; Yang, Y.; Cavallaro, A.; and Xiang, T. 2019a. Learning generalisable omni-scale representations for person re-identification. *arXiv preprint arXiv:1910.06827*.

Zhou, K.; Yang, Y.; Cavallaro, A.; and Xiang, T. 2019b. Omni-scale feature learning for person re-identification. In *ICCV*.

Zhu, J.-Y.; Park, T.; Isola, P.; and Efros, A. A. 2017. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *ICCV*.