Domain Generalization Using a Mixture of Multiple Latent Domains

Toshihiko Matsuura[1], **Tatsuya Harada**[1,2]
[1]The University of Tokyo [2]RIKEN
{matsuura, harada}@mi.t.u-tokyo.ac.jp

Abstract

When domains, which represent underlying data distributions, vary during training and testing processes, deep neural networks suffer a drop in their performance. Domain generalization allows improvements in the generalization performance for unseen target domains by using multiple source domains. Conventional methods assume that the domain to which each sample belongs is known in training. However, many datasets, such as those collected via web crawling, contain a mixture of multiple latent domains, in which the domain of each sample is unknown. This paper introduces domain generalization using a mixture of multiple latent domains as a novel and more realistic scenario, where we try to train a domain-generalized model without using domain labels. To address this scenario, we propose a method that iteratively divides samples into latent domains via clustering, and which trains the domain-invariant feature extractor shared among the divided latent domains via adversarial learning. We assume that the latent domain of images is reflected in their style, and thus, utilize style features for clustering. By using these features, our proposed method successfully discovers latent domains and achieves domain generalization even if the domain labels are not given. Experiments show that our proposed method can train a domain-generalized model without using domain labels. Moreover, it outperforms conventional domain generalization methods, including those that utilize domain labels.

Introduction

In the development of deep neural networks (DNNs), many methods that achieve good performance in computer vision tasks have been proposed (Ren et al. 2015; Chen et al. 2018). A domain represents an underlying data distribution, and these methods assume that the domains given in training (*source domain*) and in testing (*target domain*) are the same. However, it is known that DNNs suffer a drop in their performance due to domain shift (Torralba and Efros 2011).

To address this problem, extensive research has been carried out on domain generalization, which aims to train a domain-generalized model that performs well for the unseen target domain by using labeled data from multiple source domains. Considering the situation where a DNN is used for autonomous driving or robots in the real world, it is desirable to perform well under different conditions (e.g., illumination, types of objects) from the data given in training. Because we can access no samples in the target domain, domain generalization can be considered a more difficult and a more important task than domain adaptation (Long et al. 2015; Ganin and Lempitsky 2015), in which we can access labeled/unlabeled samples of the target domain in training.

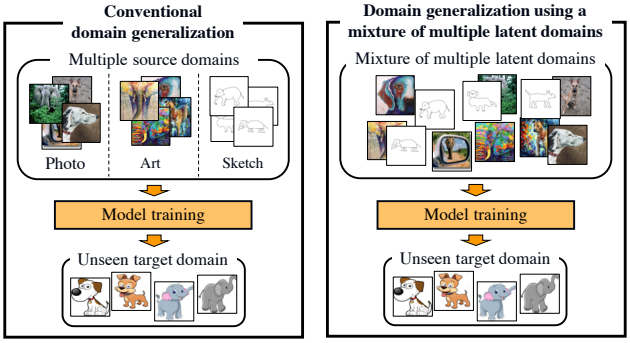

Figure 1: Unlike conventional domain generalization, domain generalization using a mixture of multiple latent domains aims to train a domain-generalized model without domain labels (e.g., Photo, Art, Sketch), which represent the domain to which each sample belongs.

To achieve domain generalization, several domain generalization methods have been proposed, including methods that train the feature extractor so that the feature distributions among multiple source domains are matched (Li et al. 2018a; Li et al. 2018b), or methods that train models for each domain and which combine them in testing (Mancini et al. 2018b; D'Innocente and Caputo 2018). These conventional methods require domain labels, which represent the domain to which each sample in multiple source domains belongs. However, most datasets, such as those collected via web crawling, are a mixture of multiple latent domains, and it is difficult to know the domain labels. For example, there are several types of image search results for "dog", such as close-up photos of a face, photos of a dog figure in na-

ture, and drawings of a dog. In this scenario, domain labels have to be attached manually to use conventional methods, but this process may be costly and time-consuming. Moreover, it is not obvious how to divide a mixture of multiple latent domains into each domain because those underlying data distributions are unknown.

JiGen (Carlucci et al. 2019) achieves domain generalization without domain labels by combining supervised learning and self-supervised learning to solve jigsaw puzzles of the training images. However, it does not take advantage of the fact that there exist several latent domains in the source domain. Therefore, in this paper, we propose a novel and realistic scenario called *domain generalization using a mixture of multiple latent domains*, in which the source domain contains multiple latent domains, and the domain to which each sample belongs is unknown. As shown in Fig. 1, in the proposed scenario, we try to train a model that performs well for the unseen target domain using a mixture of multiple latent domains. Moreover, we propose a novel method to solve this scenario. First, we assume that the latent domain of images is reflected in their style. Although other factors may also be considered, such as the background, location, and pose change, domain mismatches may be more severe when image styles are different, such as photos, NIR images, paintings, or sketches. Therefore, we utilize style features proposed in the research field of style transfer as domain-discriminative features to discover latent domains. Specifically, we utilize a stack of convolutional feature statistics (i.e., mean and standard deviation) that are known to be capable of capturing image styles (Li et al. 2017c). Once domain-discriminative features are obtained, our method iteratively assigns pseudo domain labels by clustering them, and trains a domain-invariant feature extractor shared among multiple latent domains by adversarial learning.

Experiments with benchmark datasets show that our proposed method is effective for domain generalization using a mixture of multiple latent domains, and it outperforms conventional domain generalization methods that use domain labels. Moreover, it is found that the use of pseudo domain labels obtained by clustering style features improves the classification performance compared with the use of original domain labels annotated by humans.

Related Work

Here, we explain domain adaptation and domain generalization methods. Moreover, we explain style-transfer methods because as domain-discriminative features, our proposed method utilizes style features that were originally proposed in the research field of style transfer.

Domain Adaptation

To deal with domain shift (Torralba and Efros 2011), domain adaptation and domain generalization have been studied. Domain adaptation aims to generalize a model from the source domain to the target domain with data in both domains. In unsupervised domain adaptation, several methods are employed to match the distribution in pixel space (Bousmalis et al. 2017; Chen et al. 2019) or feature space (Long et al. 2015; Ganin and Lempitsky 2015). Although these methods assume single-source and target domains, multi-source domain adaptation methods (Xu et al. 2018; Schoenauer-Sebag et al. 2019) utilize multiple source domains for domain adaptation to learn domain relations.

Moreover, for the case in which the domains to which each sample belongs are unknown, Mancini et al. (Mancini et al. 2018a) proposed a deep architecture that automatically discovers multiple latent domains, and it uses this information to align the distributions of the internal feature representations of sources and target domains. In contrast to our proposed method, this method is suitable for domain adaptation, and requires target samples in training.

Domain Generalization

Domain generalization aims to train a domain-generalized model for the unseen target domain by using multiple source domains. Unlike domain adaptation, target samples are not given in training. The representative methods for domain generalization match the feature distributions among multiple source domains by using an auto-encoder (Ghifary et al. 2015; Li et al. 2018a) or using adversarial learning (Li et al. 2018b; Shao et al. 2019). In addition, several methods have been proposed, such as a method that is based on meta learning (Li et al. 2017b; Balaji, Sankaranarayanan, and Chellappa 2018), one that uses domain-specific aggregation modules (D'Innocente and Caputo 2018), and a method that combines supervised learning and self-supervised learning to solve jigsaw puzzles (Carlucci et al. 2019).

Most conventional domain generalization methods require domain labels, which represent the domains to which each sample belongs. However, in the scenario of domain generalization using a mixture of multiple domains, we cannot apply these methods because domain labels are not given. Although JiGen (Carlucci et al. 2019) does not require domain labels in training, it is different from our proposed method, which assumes that the source domain contains multiple latent domains and take advantage of them.

Style Transfer

Style transfer enables us to transfer the style of an image called *style image* to that of an image called *content image* while preserving its content. Neural style transfer (Gatys, Ecker, and Bethge 2016) utilizes Gram matrices of the neural activations from different layers of a convolutional neural network (CNN) to represent the artistic style of an image. Li et al. (Li et al. 2017c) theoretically showed that matching the Gram matrices of neural activations is equivalent to minimizing the maximum mean discrepancy with the second-order polynomial kernel, and constructed another style loss by aligning the convolutional feature statistics (i.e., mean and standard deviation) of two feature maps between style and generated images. AdaIN (Huang and Belongie 2017) enables arbitrary style transfer in real-time by replacing the convolutional feature statistics of the content image with those of the style image. Inspired by these methods, we assume that the latent domain of images is reflected in their style and utilize convolutional feature statistics as domain-discriminative features.

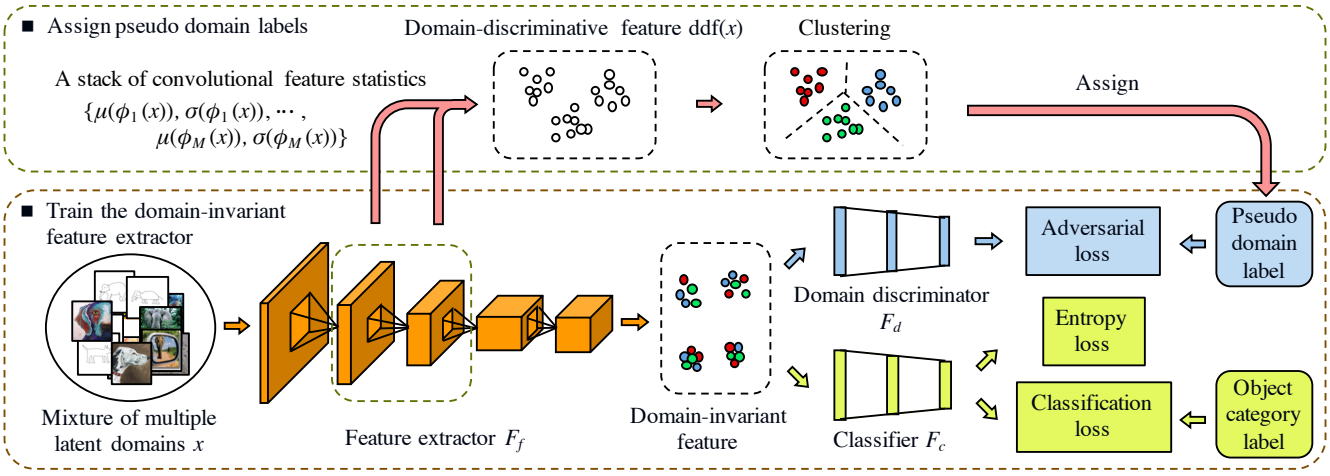

Figure 2: Illustration of our proposed method: Our method iteratively assigns pseudo domain labels by clustering domain-discriminative features extracted from lower layers of the feature extractor, and trains the domain-invariant feature extractor via adversarial learning.

Domain Generalization Using a Mixture of Multiple Latent Domains

In conventional domain generalization, the model trained with K source domains $\mathcal{D}_s = \{\mathcal{D}_s^k\}_{k=1}^K$, which share the same tasks (input x and label spaces y) but have different data distributions, accurately works for the new target domain $\mathcal{D}_t$. In this paper, we focus on the image classification task and set the number of object categories to C. Moreover, when the k-th source domain $\mathcal{D}_s^k$ has N_s^k samples, the dataset given in training is $\mathcal{D}_s = \{\mathcal{D}_s^k\}_{k=1}^K, \mathcal{D}_s^k = \{(x_i^k, y_i^k)\}_{i=1}^{N_s^k}$. This can also be represented using $\mathcal{D}_s = \{(x_i, y_i, d_i)\}_{i=1}^{N_s}$, when the domain to which each sample belongs and the total number of samples included in all source domains are defined as d_i and N_s, respectively. Namely, conventional domain generalization methods train the model that works well for the unseen target domain by using input images x_i, object category labels y_i, and domain labels d_i.

However, as we described above, a real dataset may be a mixture of multiple latent domains, and it is difficult to obtain domain labels in this case. Therefore, we propose a scenario called domain generalization using a mixture of multiple latent domains, where the given dataset is $\mathcal{D}_s = \{(x_i, y_i)\}_{i=1}^{N_s}$ because domain labels d_i are unknown.

Proposed Method

In this section, we explain the details of our proposed method. An overview of our method is shown in Fig. 2. Our method utilizes adversarial learning with a domain discriminator to train the domain-invariant feature extractor from among multiple latent domains; this approach is also used in conventional domain adaptation or generalization methods (Ganin and Lempitsky 2015; Li et al. 2018b). Although adversarial domain generalization methods require domain labels, they are not given in domain generalization using a mixture of multiple latent domains. Therefore, our method iteratively reassigns pseudo domain labels by clustering domain-discriminative features obtained from the model.

The key point is how to extract domain-discriminative features from the model in order to cluster samples by their latent domains. Clustering features obtained from the model may generally divide samples by their object categories, and not by their domains. Moreover, our method aims to train a domain-invariant feature extractor by making the outputs domain-invariant, which hinders the extraction of domain-discriminative features from the model. To solve this problem, we assume that the latent domain of images is reflected in their style, and we thus propose to utilize style features used in style transfer. Specifically, we utilize a stack of convolutional feature statistics (i.e., mean and standard deviations) obtained from lower layers of the feature extractor. In this way, our method can divide samples into each latent domain and achieve domain generalization. In the rest of the section, we describe the details of each component of our proposed method.

Adversarial Domain Generalization

Adversarial learning, which is developed from generative adversarial networks (GANs) (Goodfellow et al. 2014), has been used for research in domain adaptation (Ganin and Lempitsky 2015) and generalization (Li et al. 2018b). Generally, a deep learning model can be divided into a feature extractor F_f and a classifier F_c. These models can be trained with the following classification loss L_{cls}.

$$L_{\text{cls}} = -\frac{1}{N_s}\sum_{i=1}^{N_s}\sum_{c=1}^{C} \mathbb{1}_{[c=y_i]} \log F_c(F_f(x_i)) \qquad (1)$$

In addition to these components, adversarial learning introduces a domain discriminator F_d, which is trained to discriminate the domains when outputs of the feature extractor are inputted. Conversely, the feature extractor is trained

to extract features that make it difficult for the domain discriminator to discriminate their domains. This makes it possible to extract domain-invariant features from among multiple source domains, which generalizes the model for the unseen target domain. The adversarial loss L_{adv} is defined as follows.

$$L_{\mathrm{adv}} = -\frac{1}{N_s}\sum_{i=1}^{N_s}\sum_{k=1}^{\hat{K}} \mathbb{1}_{[k=\hat{d}_i]} \log F_d(F_f(x_i)) \tag{2}$$

Although conventional methods use known domain labels d_i and the known number of domains K, our proposed method uses pseudo domain labels $\hat{d}_i$ by assigning samples into $\hat{K}$ pseudo domains using clustering.

It is known that adversarial learning tends to generate ambiguous features near the decision boundary by trying to simply match the distributions among multiple source domains (Saito et al. 2018). Therefore, we introduce the entropy loss L_{ent} (Grandvalet and Bengio 2005), which is used in some domain adaptation methods (Long et al. 2016; Zhang et al. 2019) to train a more discriminative model for target samples by encouraging low-density separation between object categories. Although previous domain adaptation methods adapt it to only unlabeled target samples, our method adapts it to all labeled source samples as follows.

$$L_{\mathrm{ent}} = -\frac{1}{N_s}\sum_{i=1}^{N_s} H(F_c(F_f(x_i))) \tag{3}$$

Here, $H(\cdot)$ represents the entropy function. This entropy loss enables us to extract discriminative features for object categories and to improve the classification accuracy.

The total training objective is described as follows.

$$\begin{aligned} \min_{F_f, F_c} &= L_{\mathrm{cls}}(F_f, F_c) + \lambda(L_{\mathrm{ent}}(F_f, F_c) - L_{\mathrm{adv}}(F_f, F_d)) \\ \min_{F_d} &= L_{\mathrm{adv}}(F_f, F_d) \end{aligned} \tag{4}$$

Here, λ denotes the trade-off parameter to suppress the noise signal of two losses $L_{\mathrm{adv}}, L_{\mathrm{ent}}$ in the early stage of training.

Domain-discriminative Features

As domain-discriminative features, we utilize style features proposed in the style transfer (Gatys, Ecker, and Bethge 2016; Li et al. 2017c; Huang and Belongie 2017). Style transfer aims to generate a stylized image given a content image and a reference style image. Li et al. (Li et al. 2017c) proposed a new style loss L_{sty} to align the convolutional feature statistics (i.e., mean and standard deviation) between the generated image x_{gen} and the style image x_{sty} as follows.

$$\begin{aligned} L_{\mathrm{sty}} = &\sum_{m=1}^{M} \|\mu(\phi_m(x_{\mathrm{gen}})) - \mu(\phi_m(x_{\mathrm{sty}}))\|_2 + \\ &\sum_{m=1}^{M} \|\sigma(\phi_m(x_{\mathrm{gen}})) - \sigma(\phi_m(x_{\mathrm{sty}}))\|_2 \end{aligned} \tag{5}$$

Here, each $\phi_m(x)$ denotes the output in a layer used to compute the style loss, and mean $\mu(x)$ and standard deviation $\sigma(x)$ are calculated across spatial dimensions independently for each channel c.

$$\mu_c(x) = \frac{1}{HW}\sum_{h=1}^{H}\sum_{w=1}^{W} x_{chw} \tag{6}$$

$$\sigma_c(x) = \sqrt{\frac{1}{HW}\sum_{h=1}^{H}\sum_{w=1}^{W}(x_{chw} - \mu_c(x))^2 + \epsilon} \tag{7}$$

In our method, we assume that the latent domain of images is reflected in their style, and we thus utilize convolutional feature statistics as domain-discriminative features. Further, to combine multi-scale style features obtained from different convolutional layers, we define a stack of them as domain-discriminative features. Namely, the domain-discriminative feature $\mathrm{ddf}(x)$ is calculated using multiple layers' outputs $\phi_1(x), \cdots, \phi_M(x)$ as follows.

$$\mathrm{ddf}(x) = \{\mu(\phi_1(x)), \sigma(\phi_1(x)), \cdots, \mu(\phi_M(x)), \sigma(\phi_M(x))\} \tag{8}$$

Algorithm 1 Training algorithm.

Require: Data: $\mathcal{D}_s = \{(x_i^s, y_i^s)\}_{i=1}^{N_s}$
  Initialize $\hat{d}_i, \hat{d}'_i$ with zero
  while not end of epoch **do**
    Calculate $\{\mathrm{ddf}(x_i)\}_{i=1}^{N_s}$ using Eq. 8
    Obtain $\{a_i\}_{i=1}^{N_s}$ by clustering $\{\mathrm{ddf}(x_i)\}_{i=1}^{N_s}$
    Calculate $\hat{\pi}$ using Eq. 9
    Update $\hat{d}_i$ with $\hat{\pi}(a_i)$
    while not end of minibatch **do**
      Sample a minibatch of $x_i, y_i, \hat{d}_i$
      Update parameters using Eq. 4
    end while
    Update $\hat{d}'_i$ with $\hat{d}_i$
  end while

Training Procedure

After obtaining domain-discriminative features for all training samples using Eq. 8, our method divides them into $\hat{K}$ clusters by clustering, and utilizes the cluster assignments a_i as pseudo domain labels $\hat{d}_i$. We use a standard clustering algorithm, k-means (Macqueen 1967), although other clustering algorithms can be used in our method. The overall training procedure is shown in Alg. 1. Our method iteratively reassigns pseudo domain labels in training. This is because domain-discriminative features can be extracted more successfully as the training progresses. In particular, we determine that the reassignment of pseudo domain labels is conducted for each epoch.

The problem here is that clustering can divide samples into each cluster but cannot properly decide which domain label should be assigned to each cluster. If the reassigned pseudo domain labels are shifted largely with those before one epoch, it negatively impacts the training. Therefore, we use the following equation to convert the cluster assignment a_i into the pseudo domain label $\hat{d}_i$ by calculating the permutation $\hat{\pi}$ so as to maximize the rate of agreement between

the cluster assignments $\{a_i\}_{i=1}^{N_s}$ and pseudo domain labels before one epoch $\{\hat{d}'_i\}_{i=1}^{N_s}$.

$$\hat{\pi} = \arg\max_{\pi \in \Pi} \frac{1}{N_s} \sum_{i=1}^{N_s} \mathbb{1}_{[\hat{d}'_i = \pi(a_i)]} \tag{9}$$

Here, the optimal permutation $\hat{\pi}$ can be computed using the Kuhn-Munkres algorithm (Munkres 1957).

Experiments

Datasets

To evaluate our proposed method, we perform experiments using two datasets for domain generalization.[1] PACS (Li et al. 2017a) consists of four domains (i.e., Photo, Art Paintings, Cartoon, and Sketch), spanning different image styles, with seven object categories. VLCS (Torralba and Efros 2011) aggregates images of five shared object categories (bird, car, chair, dog, and person) from PASCAL VOC 2007 (Everingham et al.), LabelMe (Russell et al. 2008), Caltech-101 (Fei-Fei, Fergus, and Perona 2007), Sun09 datasets (Choi et al. 2010) which are considered as four separate domains. Unlike PACS, VLCS provides only photo images with different camera types or composition bias. By using the VLCS dataset, we verify whether our method, which focuses on the image styles, can also deal with domain shifts inside photos.

Following the previous work (Carlucci et al. 2019), we use three domains as the source domain, and the other as the target. For the same reason, we split 10% (in the case of PACS) and 30% (in the case of VLCS) of the source samples as validation datasets. In testing, all target samples are used to calculate the accuracy of the model that achieves the best accuracy in the validation dataset. Because domain labels are not given in domain generalization using a mixture of multiple latent domains, we do not use them when using our method.

Implementation Details

As the feature extractor, we use AlexNet and ResNet-18 pre-trained on ImageNet by removing the last layer. As the classifier, we initialize one fully connected layer to have the same number of inputs as before, and to have the same number of outputs as the number of object categories. As the domain discriminator, we use three fully connected layers ($1024 \to 1024 \to \hat{K}$). Note that we weight the loss function in Eq. 2 by the inverse of the size of pseudo domain labels. This is because if the number of images per pseudo domain is highly imbalanced, minimizing Eq. 2 results in a trivial parametrization where the model will predict the same output regardless of the input. To acquire the domain-discriminative features of Eq. 8, we use `relu2` and `relu3` in the case of AlexNet, and `conv2_x` and `conv3_x` in the case of ResNet-18. To conduct adversarial learning in Eq. 4, we insert a gradient reversal layer (GRL) (Ganin and Lempitsky 2015) between the feature extractor and the domain discriminator, and we use the same schedule for λ of Eq. 4 as follows: $\lambda = \frac{2}{1+\exp(-10 \cdot p)} - 1$. Here, p is linearly changed from 0 to 1 as training progresses. To reduce the computational cost of clustering, we reduce the dimension of domain-discriminative features to 256.

Basically, we utilize the other hyper-parameters employed by JiGen (Carlucci et al. 2019). In other words, we train the model for 30 epochs using the mini-batch stochastic gradient descent (SGD) with a momentum of 0.9, a weight decay of $5e-4$, and a batch size of 128. We set the initial learning rate to $1e-3$, and scale it by a factor of 0.1 after 80% of the training epochs. In the experiment with the VLCS dataset, we set the initial learning rate to $1e-4$ because it is observed that a high learning rate causes early convergence and overfitting in the source domain. Moreover, we set the learning rate of the classifier and the domain discriminator to be 10 times larger than that of the feature extractor because they are trained from scratch. For pre-processing, we crop images to random sizes and aspect ratios, horizontally flip them randomly, change their brightness/contrast/saturation/hue randomly, and normalize them using ImageNet's statistics.

Baselines

We compare our method with the following recent domain generalization methods. Deep All: Pre-trained Alexnet or ResNet-18 fine-tuned on the aggregation of all source domains with only the classification loss. TF (Li et al. 2017a): The low-rank parameterized neural network, which reduces the number of parameters to be trained. CIDDG (Li et al. 2018b): The conditional-invariant deep domain generalization method, which matches conditional distributions by considering the changes in the class prior. MLDG (Li et al. 2017b): The meta-learning method by meta-optimization on simulated train/test splits with the domain shift. CCSA (Motiian et al. 2017): The deep model in mixture with the classification and contrastive semantic alignment loss to address supervised domain adaptation and generalization. MMD-AAE (Li et al. 2018a): A model that trains feature representations by jointly optimizing a multi-domain autoencoder regularized by the maximum mean discrepancy distance, a discriminator, and a classifier with adversarial learning. SLRC (Ding and Fu 2018): The structured low-rank constraint to transfer the knowledge between domain-specific networks and the domain-invariant one. D-SAM (D'Innocente and Caputo 2018): Domain-specific aggregation modules, which enable us to merge generic and specific information in an effective manner using an aggregation layer strategy. JiGen (Carlucci et al. 2019): Jigsaw puzzle-based generalization method, which focuses on the unsupervised task to solve jigsaw puzzles.

Note that methods other than Deep All and JiGen cannot be applied for domain generalization using a mixture of multiple latent domains because they require domain labels in training. Therefore, for these methods, we use the score in the scenario of general domain generalization where domain labels are given.

[1] The code is publicly available at https://github.com/mil-tokyo/dg_mmld/.

PACS	Art.	Cartoon	Sketch	Photo	Avg.
AlexNet					
Deep All	63.30	63.13	54.07	87.70	67.05
TF*	62.86	66.97	57.51	**89.50**	69.21
Deep All	57.55	67.04	58.52	77.98	65.27
CIDDG*	62.70	69.73	64.45	78.65	68.88
Deep All	64.91	64.28	53.08	86.67	67.24
MLDG*	66.23	66.88	58.96	88.00	70.01
Deep All	64.44	72.07	58.07	87.50	70.52
D-SAM*	63.87	70.70	64.66	85.55	71.20
Deep All	66.68	69.41	60.02	89.98	71.52
JiGen	67.63	71.71	65.18	89.00	73.38
Deep All	68.09	70.23	61.80	88.86	72.25
Ours ($\hat{K}$=2)	66.99	70.64	**67.78**	89.35	73.69
Ours ($\hat{K}$=3)	**69.27**	**72.83**	66.44	88.98	**74.38**
Ours ($\hat{K}$=4)	68.84	72.53	65.90	88.75	74.01
ResNet-18					
Deep All	77.87	75.89	69.27	95.19	79.55
D-SAM*	77.33	72.43	**77.83**	95.30	80.72
Deep All	77.85	74.86	67.74	95.73	79.05
JiGen	79.42	75.25	71.35	96.03	80.51
Deep All	78.34	75.02	65.24	96.21	78.70
Ours ($\hat{K}$=2)	**81.28**	**77.16**	72.29	**96.09**	**81.83**
Ours ($\hat{K}$=3)	79.64	76.75	71.22	95.86	80.87
Ours ($\hat{K}$=4)	80.07	75.06	74.21	95.73	81.26

Table 1: Results in the PACS dataset. The title of each column indicates the name of the domain used as the target. The methods with an asterisk use domain labels, but Deep All, JiGen, and our method do not use them. The respective scores are obtained from each method's original paper.

Results

Table 1 and Table 2 show the experimental results with the PACS and VLCS datasets, respectively. The scores shown in the tables are the average over five repetitions for each run, and $\hat{K}$ denotes the number of pseudo domains used in our method. For all datasets, our method achieves results that surpass those of existing methods regardless of the number of pseudo domains $\hat{K}$. Below, we discuss the influence of the number of pseudo domains $\hat{K}$. In the PACS dataset, our method has a significant advantage with respect to the corresponding Deep All baseline. The results show that training the domain-invariant feature extractor using adversarial learning is effective for domain generalization among more diverse domains such as the PACS dataset. This good performance is achieved without using any domain labels, unlike other methods excluding JiGen. Our method can discover latent domains and assign pseudo domain labels by focusing on the image styles.

Moreover, even in the VLCS dataset, where domain shifts are inside only photo images, our method can improve the classification accuracy compared to other methods. The results show that even if the original domain labels of datasets are not separated by the image styles, our method can improve the generalization performance by assigning pseudo domain labels by focusing on them.

VLCS	Caltech	Labelme	Pascal	Sun	Avg.
AlexNet					
Deep All	85.73	61.28	62.71	59.33	67.26
CIDDG*	88.83	63.06	64.38	62.10	69.59
Deep All	86.10	55.60	59.10	54.60	63.85
CCSA*	92.30	62.10	67.10	59.10	70.15
Deep All	86.67	58.20	59.10	57.86	65.46
SLRC*	92.76	62.34	65.25	63.54	70.97
Deep All	93.40	62.11	68.41	64.16	72.02
TF*	93.63	**63.49**	69.99	61.32	72.11
MMD-AAE*	94.40	62.60	67.70	64.40	72.28
Deep All	94.45	57.45	66.06	65.87	71.08
D-SAM*	91.75	56.95	58.59	60.84	67.03
Deep All	96.63	59.18	71.96	62.57	72.66
JiGen	96.93	60.90	70.62	64.30	73.19
Deep All	95.89	57.88	72.01	67.76	73.39
Ours ($\hat{K}$=2)	96.66	58.77	71.96	**68.13**	**73.88**
Ours ($\hat{K}$=3)	**97.02**	58.37	71.40	67.89	73.67
Ours ($\hat{K}$=4)	96.57	58.66	**72.09**	66.79	73.53

Table 2: Results in the VLCS dataset. The respective scores are obtained from each method's original paper. For details about the meaning of columns and use of asterisks, see Table 1.

PACS	Art.	Cartoon	Sketch	Photo	Avg.
AlexNet					
Deep All	68.09	70.23	61.80	88.86	72.25
Ours w/o L_{adv}	67.66	70.45	62.56	88.94	72.40
Ours w/o L_{ent}	68.31	71.13	65.26	**89.38**	73.52
Ours w/o stat.	67.37	70.22	63.12	89.20	72.48
Ours w/o iter.	69.13	70.72	65.41	89.11	73.59
Ours w/o clus.	68.49	72.24	66.31	89.27	74.08
Ours	**69.27**	**72.83**	**66.44**	88.98	**74.38**

Table 3: Results of the ablation study in the PACS dataset. For details about the meaning of columns, see Table 1.

Further Analysis

Ablation Study

In this section, we describe an ablation study to investigate the effect of different components of our method using the PACS dataset and AlexNet. The variants of our method used in the experiments are as follows. Our method without L_{adv}: The model that removes the adversarial loss in Eq. 2. Our method without L_{ent}: The model that removes the entropy loss in Eq. 3. Our method without stat.: The model that simply uses outputs of the convolutional layer (`relu2` in this experiment) as domain-discriminative features for clustering instead of a stack of convolutional feature statistics in Eq 8. Our method without iter.: The model that uses the first assigned pseudo domain labels to the end without iteratively reassigning them. Our method without clus.: The model that uses original domain labels instead of assigning pseudo domain labels by clustering.

Table 3 shows the experimental results obtained when the number of pseudo domains is set to three. The results of our method without L_{adv} and our method without L_{ent} indicate that the adversarial loss in Eq. 2 is effective for domain generalization, and it is further improved by using the entropy

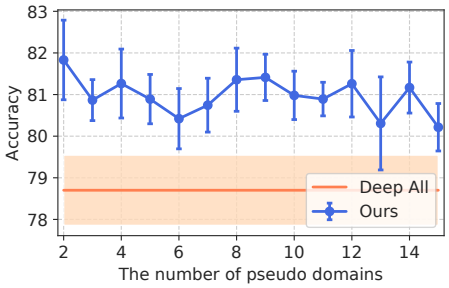

Figure 3: Results obtained when varying the number of pseudo domains. The accuracy is the average of five sets.

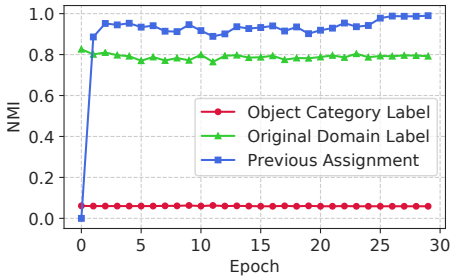

Figure 4: NMI between pseudo domain labels and object category labels, original domain labels, and previous assignments.

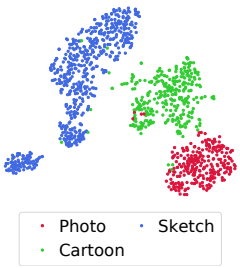

Figure 5: T-SNE visualization of domain discriminative features.

loss in Eq. 3. The result of our method without stat. indicates that simply using the outputs of convolutional layers cannot sufficiently extract domain-discriminative features, and it cannot achieve domain generalization so well. The result of our method without iter. indicates that iteratively reassigning pseudo domain labels improve the classification performance compared with those assigned at the start of training to the end. This may be because domain-discriminative features can be extracted more successfully by using models trained with samples of each domain, rather than using a pretrained model. Finally, the result of our method without clus. indicates that the use of iteratively reassigned pseudo domain labels improves the classification accuracy compared with the use of original domain labels. It appears that pseudo domain labels are suitable for training the domain-invariant feature extractor because they are based on the model's inner features and capture image styles.

Varying the Number of Pseudo Domains

In domain generalization using a mixture of multiple latent domains, the number of multiple latent domains in the source domain is unknown. Although our method divides samples into $\hat{K}$ pseudo domains by clustering, we have to set the number of pseudo domains in advance. It is unclear whether our method works accurately if the number of pseudo domains is not the same as the number of original domains. Therefore, we check the performance of our method when changing the number of pseudo domains. We use the same experimental setting of the previous paragraph with the PACS dataset and ResNet-18. We consider four experiments in which the target domains are changed as one set, and repeat it five times. Fig. 3 shows the mean and standard deviation results of our proposed method and Deep All. Note that in reality, the number of original domains is three. Based on the results obtained, there is no significant correlation between the number of pseudo domains and the classification accuracy, which highlights the robustness of our method to varying numbers of pseudo domains.

Clustering Evaluation

Our method assigns pseudo domain labels by clustering. There is a concern that clustering is not performed by domains but by object categories, although it does not necessarily have to divide samples by original domains. Therefore, we evaluate the clustering by calculating the normalized mutual information (NMI) between pseudo domain labels and object category labels, original domain labels, and pseudo domain labels before one epoch. Moreover, we visualize the distribution of domain-discriminative features using t-SNE (van der Maaten and Hinton 2008). We use the same experiment setting of the previous paragraph with the PACS dataset and AlexNet, set the number of pseudo domains to three, and set Art-painting to the target domain.

Fig. 4 shows that the NMI between pseudo domain labels and the original domain labels is large, while that between pseudo domain labels and object category labels is small. Moreover, the NMI between pseudo domain labels and original domain labels remains almost unchanged over the whole training period. These indicate that clustering domain-discriminative features divides samples not by object categories but original domains over the whole training period. This fact can also be seen in Fig. 5, where the distributions of domain-discriminative features are roughly divided by their original domains. Moreover, the NMI between pseudo domain labels and the previous assignment gradually converges to 1.0 as the training proceeds, which indicates that clustering results become gradually stable.

Conclusion

In this study, we proposed a new scenario called domain generalization using a mixture of multiple latent domains. To address this scenario, we proposed a new method that extracts a stack of convolutional feature statistics representing the image styles as domain-discriminative features, assigns pseudo domain labels by clustering them, and trains the domain-invariant feature extractor from among latent domains using adversarial learning. In the experiments, our method without domain labels achieved a better performance than conventional methods that use them.

Acknowledgments

This work was partially supported by JST CREST Grant Number JPMJCR1403, and partially supported by JSPS KAKENHI Grant Number JP19H01115. We would like to thank Yusuke Mukuta, Antonio Tejero de Pablos, Atsuhiro Noguchi, Akihiro Nakamura for helpful discussions.

References

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

[Bousmalis et al. 2017] Bousmalis, K.; Silberman, N.; Dohan, D.; Erhan, D.; and Krishnan, D. 2017. Unsupervised pixel-level domain adaptation with generative adversarial networks.

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

[Chen et al. 2018] Chen, L.-C.; Zhu, Y.; Papandreou, G.; Schroff, F.; and Adam, H. 2018. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *ECCV*.

[Chen et al. 2019] Chen, Y.-C.; Lin, Y.-Y.; Yang, M.-H.; and Huang, J.-B. 2019. Crdoco: Pixel-level domain transfer with cross-domain consistency. In *CVPR*.

[Choi et al. 2010] Choi, M. J.; Lim, J. J.; Torralba, A.; and Willsky, A. S. 2010. Exploiting hierarchical context on a large database of object categories. In *CVPR*.

[Ding and Fu 2018] Ding, Z., and Fu, Y. 2018. Deep domain generalization with structured low-rank constraint. *IEEE Transactions on Image Processing* 27:304–313.

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

[Everingham et al.] Everingham, M.; Van Gool, L.; Williams, C. K. I.; Winn, J.; and Zisserman, A. The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results. http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html.

[Fei-Fei, Fergus, and Perona 2007] Fei-Fei, L.; Fergus, R.; and Perona, P. 2007. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. *Computer Vision and Image Understanding* 106(1):59–70.

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

[Gatys, Ecker, and Bethge 2016] Gatys, L. A.; Ecker, A. S.; and Bethge, M. 2016. Image style transfer using convolutional neural networks. In *CVPR*.

[Ghifary et al. 2015] Ghifary, M.; Kleijn, W. B.; Zhang, M.; and Balduzzi, D. 2015. Domain generalization for object recognition with multi-task autoencoders. In *ICCV*.

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

[Grandvalet and Bengio 2005] Grandvalet, Y., and Bengio, Y. 2005. Semi-supervised learning by entropy minimization. In *NIPS*.

[Huang and Belongie 2017] Huang, X., and Belongie, S. 2017. Arbitrary style transfer in real-time with adaptive instance normalization. In *ICCV*.

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

[Li et al. 2017b] Li, D.; Yang, Y.; Song, Y.-Z.; and Hospedales, T. M. 2017b. Learning to generalize: Meta-learning for domain generalization. In *AAAI*.

[Li et al. 2017c] Li, Y.; Wang, N.; Liu, J.; and Hou, X. 2017c. Demystifying neural style transfer. In *IJCAI*.

[Li et al. 2018a] Li, H.; Jialin Pan, S.; Wang, S.; and Kot, A. C. 2018a. Domain generalization with adversarial feature learning. In *CVPR*.

[Li et al. 2018b] Li, Y.; Tian, X.; Gong, M.; Liu, Y.; Liu, T.; Zhang, K.; and Tao, D. 2018b. Deep domain generalization via conditional invariant adversarial networks. In *ECCV*.

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

[Long et al. 2016] Long, M.; Zhu, H.; Wang, J.; and Jordan, M. I. 2016. Unsupervised domain adaptation with residual transfer networks. In *NIPS*.

[Macqueen 1967] Macqueen, J. 1967. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley Symposium on Mathematical Statistics and Probability*, 281–297.

[Mancini et al. 2018a] Mancini, M.; Porzi, L.; Rota Bulò, S.; Caputo, B.; and Ricci, E. 2018a. Boosting domain adaptation by discovering latent domains. In *CVPR*.

[Mancini et al. 2018b] Mancini, M.; Rota Bulò, S.; Caputo, B.; and Ricci, E. 2018b. Best sources forward: Domain generalization through source-specific nets. In *ICIP*.

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

[Munkres 1957] Munkres, J. 1957. Algorithms for the assignment and transportation problems. *Journal of the Society for Industrial and Applied Mathematics* 5:32–38.

[Ren et al. 2015] Ren, S.; He, K.; Girshick, R.; and Sun, J. 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NIPS*.

[Russell et al. 2008] Russell, B. C.; Torralba, A.; Murphy, K. P.; and Freeman, W. T. 2008. Labelme: A database and web-based tool for image annotation. *International Journal of Computer Vision* 77(1-3):157–173.

[Saito et al. 2018] Saito, K.; Watanabe, K.; Ushiku, Y.; and Harada, T. 2018. Maximum classifier discrepancy for unsupervised domain adaptation. In *CVPR*.

[Schoenauer-Sebag et al. 2019] Schoenauer-Sebag, A.; Heinrich, L.; Schoenauer, M.; Sebag, M.; Wu, L.; and Altschuler, S. 2019. Multi-domain adversarial learning. In *ICLR*.

[Shao et al. 2019] Shao, R.; Lan, X.; Li, J.; and Yuen, P. C.

2019. Multi-adversarial discriminative deep domain generalization for face presentation attack detection. In *CVPR*.

[Torralba and Efros 2011] Torralba, A., and Efros, A. A. 2011. Unbiased look at dataset bias. In *CVPR*.

[van der Maaten and Hinton 2008] van der Maaten, L., and Hinton, G. 2008. Visualizing data using t-SNE. *JMLR* 9:2579–2605.

[Xu et al. 2018] Xu, R.; Chen, Z.; Zuo, W.; Yan, J.; and Lin, L. 2018. Deep cocktail network: Multi-source unsupervised domain adaptation with category shift. In *CVPR*.

[Zhang et al. 2019] Zhang, Y.; Tang, H.; Jia, K.; and Tan, M. 2019. Domain-symmetric networks for adversarial domain adaptation. In *CVPR*.