Deep Domain Generalization via Conditional Invariant Adversarial Networks

Ya Li[1], Xinmei Tian[1(✉)], Mingming Gong[2,3], Yajing Liu[1], Tongliang Liu[4], Kun Zhang[2], and Dacheng Tao[4]

[1] CAS Key Laboratory of Technology in Geo-Spatial Information Processing and Application Systems, University of Science and Technology of China, Hefei, China
{muziyiye,lyj123}@mail.ustc.edu.cn, xinmei@ustc.edu.cn
[2] Department of Philosophy, Carnegie Mellon University, Pittsburgh, USA
gongmingnju@gmail.com, kunz1@cmu.edu
[3] Department of Biomedical Informatics, University of Pittsburgh, Pittsburgh, USA
[4] UBTECH Sydney AI Centre, SIT, FEIT, University of Sydney, Sydney, Australia
tliang.liu@gmail.com, dacheng.tao@sydney.edu.au

Abstract. Domain generalization aims to learn a classification model from multiple source domains and generalize it to unseen target domains. A critical problem in domain generalization involves learning domain-invariant representations. Let X and Y denote the features and the labels, respectively. Under the assumption that the conditional distribution $P(Y|X)$ remains unchanged across domains, earlier approaches to domain generalization learned the invariant representation $T(X)$ by minimizing the discrepancy of the marginal distribution $P(T(X))$. However, such an assumption of stable $P(Y|X)$ does not necessarily hold in practice. In addition, the representation learning function $T(X)$ is usually constrained to a simple linear transformation or shallow networks. To address the above two drawbacks, we propose an end-to-end conditional invariant deep domain generalization approach by leveraging deep neural networks for domain-invariant representation learning. The domain-invariance property is guaranteed through a conditional invariant adversarial network that can learn domain-invariant representations w.r.t. the joint distribution $P(T(X), Y)$ if the target domain data are not severely class unbalanced. We perform various experiments to demonstrate the effectiveness of the proposed method.

Keywords: Domain generalization · Adversarial networks
Domain invariant representation

V. Ferrari et al. (Eds.): ECCV 2018, LNCS 11219, pp. 647–663, 2018.
https://doi.org/10.1007/978-3-030-01267-0_38

1 Introduction

With the advances in deep learning in recent years, computer vision has achieved impressive success, e.g., in image classification [1,2], face recognition [3,4], and object detection [5]. The mentioned tasks rely on standard supervised learning that assumes that the training and test data follow the same distribution. However, this assumption does not hold in many real-world situations due to various changing factors, such as background noise, viewpoint changes and illumination variation. Such factors can cause biases in the collected datasets [6]. In deep learning, a common approach to eliminating data bias is through fine-tuning a pre-trained network on the target domain with a certain number of labels. However, labeling the data when moving to different new target domains is labor intensive. Domain generalization [7–16] is proposed to overcome such challenges. Given inputs and the corresponding outputs of multiple source domains, domain generalization aims to learn a domain-invariant feature representation that can generalize well to unseen target domains.

Most existing domain generalization methods learn the invariant feature transformation based on handcrafted features or features extracted from pre-trained deep learning models. Compared to handcrafted features, features extracted from pre-trained neural networks are more discriminative and descriptive. Several domain generalization methods [7–11] have demonstrated the effectiveness of features extracted from neural networks. However, the referenced methods consider the extracted features as input X and use a linear transformation or multilayer perceptrons to model the transformation $T(X)$. Such a learning strategy does not fully explore the advantages of deep neural networks. We argue that learning the invariant feature transformation directly from the original image in an end-to-end fashion will lead to better performance.

In addition, previous studies assume that the conditional distribution $P(Y|X)$ remains stable across domains and that domain-invariant learning boils down to the guarantee of invariance of the marginal distribution $P(T(X))$. If this assumption is violated, the joint distribution $P(T(X), Y)$ will not be invariant even if $P(T(X))$ is invariant after learning. According to recent results in causal learning [17,18], if the causal structure is $X \rightarrow Y$, where X is the cause and Y is the effect, $P(Y|X)$ can remain stable as $P(X)$ changes because they are "independent" of each other. However, the causal structure is often $Y \rightarrow X$ in computer vision, e.g., object classes are the causes of image features [19]. In this scenario, if $P(X)$ changes, $P(Y|X)$ often changes together with $P(X)$. Considering digital number classification as an example and denoting each rotation angle α as one domain, we obtain a different class-conditional distribution $P(X|Y, \alpha = \alpha_i)$ for each domain, i.e., the feature distribution of digital numbers depends on the rotation angle. Assuming, for simplicity, that $P(Y)$ does not change, according to the sum rule, we obtain $P(X|\alpha = \alpha_i) = \sum_{j=1}^{L} P(X|Y = j, \alpha = \alpha_i)P(Y = j)$, where L is the number of classes, and thus, the values of $P(X|\alpha = \alpha_i)$ are different across domains. Additionally, according to Bayes' rule, $P(Y|X, \alpha = \alpha_i) = P(X|Y, \alpha = \alpha_i)P(Y)/P(X|\alpha = \alpha_i)$; hence, it is very unlikely that $P(Y|X, \alpha = \alpha_i)$ are the same across domains.

In this paper, we consider the scenario whereby both $P(X)$ and $P(Y|X)$ can change across domains and address domain generalization in an end-to-end deep learning framework. This is achieved by learning a conditional invariant neural network that learns to minimize the discrepancy in $P(X|Y)$ across different domains. Inspired by generative adversarial networks [20] and recent deep domain adaptation methods [21,22], we develop an adversarial network to learn a domain-invariant representation by making the learned representations on different domains indistinguishable. The conditional invariance property is guaranteed by two adversarial losses that consider the source-domain label information overlooked by the existing methods. One aims to directly make the representations in each class indistinguishable across source domains. The other loss aims to make the representations of all classes indistinguishable across class prior-normalized source domains. The purpose of introducing class prior-based normalization is to reduce the negative effect caused by the possible class prior $P(Y)$ changes across source domains. If the prior distributions $P(Y)$ in the target domain and the pooled source domains are identical, our method can guarantee the invariance of the joint distribution $P(X, Y)$ across domains.

2 Related Work

Domain generalization has drawn substantial attention in recent years, with various approaches [7–11] having been proposed. Muandet et al. [9] proposed a domain-invariant component analysis that learns an invariant transformation by minimizing dissimilarity among domains. Ghifary et al. [8] proposed a unified framework for domain adaptation and generalization using scatter component analysis. In contrast to the above methods, Khosla et al. [7] proposed removing the dataset bias by measuring the dataset-specific model as a combination of the dataset-specific bias and a visual world model. Considering the construction ability of an autoencoder, Ghifary et al. [11] proposed a multi-task autoencoder method to learn domain-invariant features. The learned features could subsequently be used as the input to classifiers. All referenced methods are shallow domain generalization methods that need handcrafted features or features extracted from pre-trained deep learning models. Note that the multi-task autoencoder uses only one hidden layer built on the pre-learned deep features. Such pre-extracted features dramatically constrain the learning ability of the existing domain generalization methods. Our method learns the domain-invariant representation from the original images in an end-to-end deep learning framework.

In addition to the shallow architecture, the existing methods assume that $P(Y|X)$ remains invariant across domains and only aim to learn a domain-invariant feature transformation $T(X)$ to guarantee the invariance of feature distribution $P(T(X))$. Recent studies of domain adaptation have noted the importance of matching joint distributions instead of the marginal distribution. [23] and [24] suggested considering the domain adaptation problem in the generalized target shift scenario, where the causal direction is $Y \rightarrow X$. In this scenario,

both the change of distribution $P(Y)$ and conditional distribution $P(X|Y)$ are considered to reduce the data bias across domains. [22,25] proposed iterative methods for matching the joint distribution by using the predicted labels from previous iterations as pseudo-labels. [26] proposed an optimal transport-based approach to matching joint distributions and obtained promising results. In contrast to the domain adaptation methods, domain generalization does not require unlabeled data from the target domains.

3 Conditional Invariant Deep Domain Generalization

3.1 Domain Generalization

Suppose the feature and label spaces are represented by $\mathcal{X}$ and $\mathcal{Y}$, respectively. A domain is represented by a joint distribution $P(X, Y)$ defined on $\mathcal{X} \times \mathcal{Y}$. To simplify notation, the m-th domain $P^m(X, Y)$ is denoted P^m, and the marginal distribution $P^m(X)$ is denoted P_X^m. In each domain, we have a sample $\mathcal{D}_m = \{(x_i^m, y_i^m)\}_{i=1}^{N^m}$ drawn from $P^m(X, Y)$, where N^m is the sample size in the m-th domain, while $(x_i^m, y_i^m) \sim P^m(X, Y)$ denotes the i-th data point in the m-th domain. Given C related source domains $\{P^1, P^2, ..., P^C\}$ and their corresponding datasets $\mathcal{D}_m = \{(x_i^m, y_i^m)\}_{i=1}^{N^m}$, where $m = \{1, 2, ..., C\}$, the goal of domain generalization is to learn a model $f : \mathcal{X} \to \mathcal{Y}$ that can well fit an unseen, yet related, target domain $P^t(X, Y)$ using all data from the source domains.

3.2 Domain Divergence

We first introduce the Jensen-Shannon divergence (JSD) that measures similarities among multiple distributions [27]. We use the marginal distribution $P(X)$ as an example to illustrate the general results in this section. The JSD among distributions $\{P^1(X), P^2(X), \ldots, P^C(X)\}$ is defined as the average of KL-divergences of each distribution from the average distribution:

$$JSD(P_X^1, \ldots, P_X^C) = \frac{1}{C} \sum\nolimits_{m=1}^{C} KL(P_X^m || \bar{P}_X), \tag{1}$$

where $\bar{P}_X = \frac{1}{C} \sum_{m=1}^{C} P_X^m$ is the average (centroid) of these distributions. In [20], a two-player minimax approach is proposed for learning a generative adversarial network and is proven to be equivalent to minimizing JSD between the generative distribution and data distribution.

We extend the two-player minimax approach to multiple players and prove its equivalence to minimizing the JS divergence among multiple distributions. Denote the distributions after a feature transformation T as $\{P_T^1(T(X)), P_T^2(T(X)), \ldots, P_T^C(T(X))\}$, or simply as $\{P_T^1, P_T^2, \ldots, P_T^C\}$. Suppose that D is the learned discriminator and $D^m(T(X))$ denotes the prediction probability with discriminator D that $T(X)$ comes from the m-th domain

P_T^m, $m \in \{1, 2, \ldots, C\}$. We define the following multi-player minimax game with value function $V(T, D^1, \ldots, D^C) = \sum_{m=1}^{C} \mathbb{E}_{x \sim P^m(x)} \log D^m(T(x))$:

$$\min_{T} \max_{D^1, D^2, \ldots, D^C} V(T, D^1, \ldots, D^C),$$
$$\text{s.t.} \sum_{m=1}^{C} D^m(T(x)) = 1. \tag{2}$$

In what follows, we will show that the above minimax game reaches a global optimum at $P_T^1 = P_T^2 = \ldots = P_T^C$, i.e., the multi-player minimax game is able to learn invariant feature representations. The following proposition provides the optimal discriminator under a fixed transformation T.

Proposition 1. *Let $x' = T(x)$ for a fixed transformation T; the optimal prediction probabilities $\{D_T^1, \ldots, D_T^C\}$ of discriminator D are*

$$D_T^{m*}(x') = P_T^m(x') / \sum_{m=1}^{C} P_T^m(x'). \tag{3}$$

Proof. For a fixed T, Eq. (2) reduces to maximizing $V(T, D^1, \ldots, D^C)$ w.r.t. $\{D^1, \ldots, D^C\}$:

$$\{D_T^{1*}, \ldots, D_T^{C*}\} = \arg\max_{D^1, \ldots, D^C} \sum_{m=1}^{C} \int_{x'} P_T^m(x') log(D^m(x')) dx'$$
$$\text{s.t.} \sum_{m=1}^{C} D^m(x') = 1. \tag{4}$$

Maximizing the value function pointwise and applying Lagrange multipliers, we obtain the following problem:

$$\{D_T^{1*}, \ldots, D_T^{C*}\} = \arg\max_{D^1, \ldots, D^C} \sum_{m=1}^{C} P_T^m(x') log(D^m(x')) + \lambda(\sum_{m=1}^{C} D^m(x') - 1).$$

Setting the derivative of the above equation w.r.t. $D^m(x')$ to zero, we obtain $D_T^{m*}(x') = -\frac{P_T^m(x')}{\lambda}$. We can solve for the Lagrange multiplier λ by substituting $D_T^{m*}(x') = -\frac{P_T^m(x')}{\lambda}$ into the constraint $\sum_{m=1}^{C} D^m(x') = 1$ to obtain $\lambda = -\sum_{m=1}^{C} P_T^m(x')$. Thus, we obtain the optimal solution $D_T^{m*}(x') = \frac{P_T^m(x')}{\sum_{m=1}^{C} P_T^m(x')}$.

Theorem 1. *If $U(T)$ is the maximum value of $V(T, D^1, \ldots, D^C)$*

$$U(T) = \sum_{m=1}^{C} \mathbb{E}_{x \sim P_T^m(x')} \Big[\log \frac{p_T^m(x')}{\sum_{m=1}^{C} P_T^m(x')} \Big], \tag{5}$$

the global minimum of the multi-player minimax game is attained if and only if $P_T^1 = P_T^2 = \ldots = P_T^C$. At this point, $U(T)$ attains the value $-C \log C$.

Proof. If we add $C \log C$ to $U(T)$, we obtain

$$\begin{aligned} U(T) + C\log C &= \sum_{m=1}^{C} \{\mathbb{E}_{x\sim P_T^m(x')}\Big[\log \frac{p_T^m(x')}{\sum_{m=1}^{C} P_T^m(x')}\Big] + \log C\} \\ &= \sum_{m=1}^{C} \mathbb{E}_{x\sim P_T^m(x')}\Big[\log \frac{p_T^m(x')}{\frac{1}{C}\sum_{m=1}^{C} P_T^m(x')}\Big] \\ &= \sum_{m=1}^{C} KL\Big(P_T^m(x')\Big|\Big|\frac{1}{C}\sum_{m=1}^{C} P_T^m(x')\Big). \end{aligned} \tag{6}$$

By using the definition of the JS divergence in Eq. (1), we obtain $U(T) = -ClogC + C \cdot JSD(P_T^1, \ldots, P_T^C)$. As the Jensen-Shannon divergence among multiple distributions is always non-negative and zero iff they are equal, we have shown that $U^* = -C \log C$ is the global minimum of $U(T)$ and that the only solution is $P_T^1 = P_T^2 = \ldots = P_T^C$, i.e., the learned feature representations on all source domains are perfectly matched.

3.3 Proposed Approach

The existing methods proposed matching the marginal distribution $P(T(X))$ across domains; however, the invariance of $P(Y|T(X))$ could not be guaranteed. Our approach corrects the changes in $P(Y|X)$ by correcting the changes in $P(X|Y)$. In the ideal scenario, we expect the deep network to learn a conditional invariant feature transformation such that $P^{m=i}(T(X)|Y) = P^{m=j}(T(X)|Y) = P^t(T(X)|Y)$, where $i, j \in \{1, 2, ..., C\}$, and P^t is a single target domain. With the conditional invariant feature transformation, we can merge all source domains into a single new domain that has the joint distribution $P^{new}(T(X), Y) = P(T(X)|Y)P^{new}(Y)$. While training on the transformed and merged source domain data, we correct the possible class imbalances so that $P^{new}(Y)$ is the same for all classes. Thus, if the target domain data are class balanced, the equality of joint distributions $P(T(X), Y)$ between source domains and target domain can be guaranteed. Even if the target domain data are class unbalanced, our method can still provide reliable results if the features and labels are highly correlated, as the class prior distribution is not important to classification in this case.

The conditional invariance property is achieved by applying the minimax game to different aspects of the distributions on the source domains, resulting in the class-conditional minimax value and class prior-normalized marginal minimax value. In the following section, we will show that such two regularization terms can be easily implemented through variants of softmax loss.

Class-conditional Minimax Value. Suppose that there are L different classes in each domain, and denote by $x_{i\sim j}^m$ an example from the j-th class in the m-th domain. The class-conditional minimax value for class j can be formulated as $V_{con}(T, D_j^1, \ldots, D_j^C) = \sum_{m=1}^{C} \mathbb{E}_{x\sim P^m(x|y=j)} \log D_j^m(T(x))$, where D_j is the

discriminator for the j-th image class. The multi-player minimax game is

$$\min_{T} \max_{D_j^1,\ldots,D_j^C} V_{con}(T, D_j^1, \ldots, D_j^C),$$
$$\text{s.t.} \sum\nolimits_{m=1}^{C} D_j^m(T(x)) = 1. \tag{7}$$

The empirical minimax game value can be formulated as follows:

$$\tilde{V}_{con}(T, D_j^1, \ldots, D_j^C) = \sum\nolimits_{m=1}^{C} \frac{1}{N_j^m} \sum\nolimits_{i=1}^{N_j^m} logD_j^m(T(x_{i\sim j}^m)), \tag{8}$$

where N_j^m denotes the number of examples in the j-th class of the m-th domain. This term computes the minimax game value among $P(X|Y)$ locally. In practice, we compute the minimax game values for all classes separately and subsequently sum such values. By optimizing the above minimax value, we can guarantee the invariance of class-conditional distributions $P(T(X)|Y = j)$ among domains.

Class Prior-Normalized Marginal Minimax Value. If the sample size is not large, overfitting can easily occur in a deep network due to a very large number of parameters. As the number of examples in certain classes is sometimes small, learning with the above class-conditional minimax value can result in overfitting. To improve learning of domain-invariant features, we propose learning a class prior-normalized marginal term that applies the minimax game value to all conditional distributions globally. Note that the marginal distribution of feature representations on the m-th domain can be formulated as

$$P^m(T(X)) = \sum\nolimits_{j=1}^{L} P^m(T(X)|Y = j)P^m(Y = j). \tag{9}$$

The above equation shows that the marginal distribution $P^m(T(X)$ is determined by the conditional distribution $P^m(T(X)|Y = j)$ and the class prior distribution $P^m(Y = j)$, where $j \in \{1, 2, ..., L\}$. As shown in [23,24], we may be able to determine the conditional invariant representation $T(X)$ by matching the marginal distribution $P(T(X))$ across domains, i.e., the invariance of $P(T(X))$ may induce invariance of $P(T(X)|Y)$ if $P(Y)$ is invariant. If $P(Y)$ also changes, even with an invariant $P(T(X)|Y)$ across domains, $P(T(X))$ can still vary according to Eq. (9). In this case, minimizing the discrepancy in $P(X)$ may lead to removal of useful information, as the effect of changing $P(Y)$ is not supposed to be corrected by learning an invariant representation from X. To remove the effect caused by the changing class prior distribution $P(Y)$, we propose normalizing the class prior distribution as $P_N^m(T(X)) = \sum_{j=1}^{L} P^m(T(X)|Y = j)\frac{1}{L}$. The above class prior-normalized distribution P_N^m enforces the prior probability for each class to be the same. Consequently, the invariant class conditional distribution across domains can guarantee equality of class prior-normalized marginal distributions across domains. Suppose that $\beta^m(Y)$ is the normalized weight to ensure that $P_N^m(T(X)) = \sum_{j=1}^{L} P^m(T(X)|Y = j)P^m(Y = j)\beta^m(Y = j) =$

$\sum_{j=1}^{L} P^m(T(X)|Y=j)\frac{1}{L}$. We apply the minimax game according to the class prior-normalized marginal distribution as follows:

$$
\begin{aligned}
&\min_{T} \max_{D^1,\ldots,D^C} V_{norm}(T, D^1, \ldots, D^C) \\
=&\min_{T} \max_{D^1,\ldots,D^C} \sum_{m=1}^{C} \mathbb{E}_{x\sim P_N^m(x)} \log D^m(T(x)) \\
=&\min_{T} \max_{D^1,\ldots,D^C} \sum_{m=1}^{C} \mathbb{E}_{x\sim \int P^m(x|y)P^m(y)\beta^m(y)dy} \log D^m(T(x)) \\
=&\min_{T} \max_{D^1,\ldots,D^C} \sum_{m=1}^{C} \int P^m(x|y)P^m(y)\beta^m(y)dy \log D^m(T(x))dx \\
=&\min_{T} \max_{D^1,\ldots,D^C} \sum_{m=1}^{C} \int P^m(x,y) \log D^m(T(x))\beta^m(y)dxdy \\
&\text{s.t.} \sum_{m=1}^{C} D^m(T(x)) = 1.
\end{aligned} \tag{10}
$$

The empirical version of a class prior-normalized minimax value is as follows:

$$
\tilde{V}_{norm}(T, D^1, \ldots, D^C) = \sum_{m=1}^{C} \frac{1}{N^m} \sum_{i=1}^{N^m} \log D^m(T(x_i^m))\beta^m(y_i^m), \tag{11}
$$

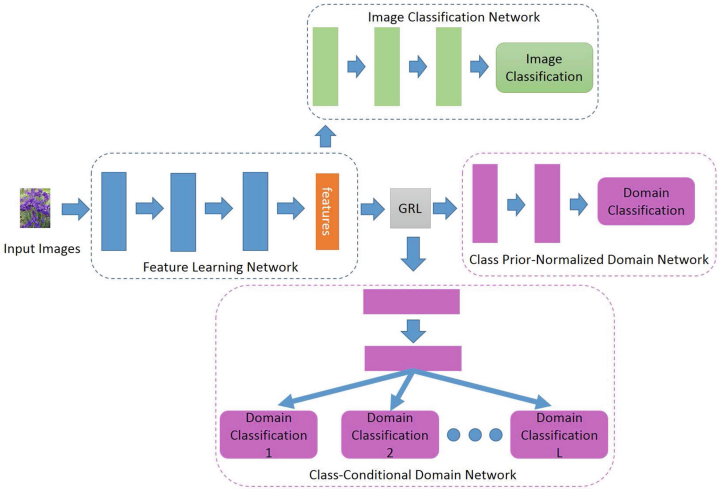


Fig. 1. The network architecture of our proposed method. It consists of four parts: feature learning network which represents the invariant feature transformation T, image classification network which classifies the images from all domains with softmax loss, class prior-normalized domain network which discriminates different domains with loss in Eq. (14), and class-conditional domain network which discriminates domains for each image class with loss in Eq. (13).

i.e., the class prior-normalized weight $\beta^m(y_i^m)$ can be viewed as a weight of the log-likelihood. And $\beta^m(y_i^m)$ can be empirically obtained as

$$\beta^m(y_i^m) = \frac{1}{L}\frac{1}{P^m(Y = y_i^m)} = \frac{N^m}{L \times N^m_{j=y_i^m}}, \tag{12}$$

where N^m denotes the total number of examples in the m-th domain and $N^m_{j=y_i^m}$ denotes the number of examples with the same label as y_i^m in the m-th domain.

4 Conditional Invariant Adversarial Network

We introduce the conditional invariant deep neural network to represent the feature transformation T and then implement the approach proposed in Sect. 3.3. The architecture is shown in Fig. 1. It contains four components: the representation learning network, the image classification network, the class-conditional domain network, and the class prior-normalized domain network. The representation learning network aims to learn a class-conditional domain-invariant feature representation, while retaining the ability to discriminate among different image classes. The two domain classification networks aim to make the features of examples from different domains indistinguishable by adversarial training. Additionally, the image classification network is used to make the learned features informative for classification. In this section, we will introduce each network and describe the process of training such networks using various loss functions.

Let x_i be an input image, $F(\cdot|\theta)$ denote a network with parameter θ, and $F(x_i|\theta)$ be the output of image x_i. To simplify notation, the feature representation learning network is denoted $F(\cdot|\theta_f)$ or $F_f(\cdot)$, the image classification network is denoted $F(\cdot|\theta_c)$ or $F_c(\cdot)$, and the class-conditional domain network for image class j is denoted $F^j(\cdot|\theta_d)$ or $F_d^j(\cdot)$. Additionally, the class prior-normalized domain network is denoted $F(\cdot|\theta_p)$ or $F_p(\cdot)$.

4.1 Class-Conditional Domain Classification Network

According to Eq. (7), we can implement the class-conditional minimax game value through a variant of softmax loss. For image class j, the class-conditional domain loss can be formulated as follows:

$$L_{con}(\theta_f, \theta_d^j) = \sum_{m=1}^{C}[\frac{1}{N_j^m}\sum_{i=1}^{N_j^m}\sum_{n=1}^{C} I[y^d_{i\sim j} = n] log P_n(F_d^j(F_f(x^m_{i\sim j})))], \tag{13}$$

where $y^d_{i\sim j} \in \{1, 2, ..., C\}$ denotes the domain label of $x_{i\sim j}$ (i-th example in class j). $P_n(F_d^j(F_f(x_{i\sim j})))$ denotes the predicted probability that the image in j-th category belongs to the n-th domain. Note that the above loss is specifically for the j-th image class. If we have L classes, we must construct L sub-branches of the networks. Each sub-branch corresponds to one class.

4.2 Class Prior-Normalized Domain Classification Network

We introduce the class prior-normalized domain classification networks according to Eq. (11). It is also implemented using a variant of softmax loss. We obtain the prior-normalized loss as

$$L_{norm}(\theta_f, \theta_p) = \sum_{m=1}^{C} \frac{1}{N^m} [\sum_{i=1}^{N^m} \sum_{n=1}^{C} \beta^m(y_i^m) I[y_i^d = n] log P_n(F_p(F_f(x_i)))], \quad (14)$$

where y_i^m denotes the class label of the i-th image in domain m.

4.3 Learning Procedure

We combine all the above losses with the image classification loss $L_{cla}(\theta_f, \theta_c)$ used for image classification networks. Note that $L_{cla}(\theta_f, \theta_c)$ can be a standard softmax loss. The total loss can be obtained as follows:

$$R(\theta_f, \{\theta_d^j\}_{j=1}^L, \theta_p, \theta_c) = L_{cla}(\theta_f, \theta_c) + \lambda(\sum_{j=1}^{L} L_{con}(\theta_f, \theta_d^j) + L_{norm}(\theta_f, \theta_p))). \quad (15)$$

The learning of the above loss can be separated into two steps by determining the optimal values $(\theta_f^*, \{\theta_d^{*j}\}_{j=1}^L, \theta_p^*, \theta_c^*)$ as follows:

$$\begin{aligned} (\theta_f^*, \theta_c^*) &= \arg\min_{\theta_f, \theta_c} R(\theta_f, \{\theta_d^j\}_{j=1}^L, \theta_p, \theta_c), \\ (\{\theta_d^{*j}\}_{j=1}^L, \theta_p) &= \arg\max_{\{\theta_d^j\}_{j=1}^L, \theta_p} R(\theta_f, \{\theta_d^j\}_{j=1}^L, \theta_p, \theta_c). \end{aligned} \quad (16)$$

A saddle point of the above optimization problem can be determined by performing the following gradient updates iteratively until the networks converge:

$$\begin{aligned} \theta_f^{i+1} &= \theta_f^i - \gamma[\frac{\partial L_{cla}^i}{\partial \theta_f} + \lambda(\sum\nolimits_{j=1}^{L} \frac{\partial L_{con}^i(\theta_f, \theta_d^j)}{\partial \theta_f} + \frac{\partial L_{norm}^i}{\partial \theta_f})], \\ \theta_c^{i+1} &= \theta_c^i - \gamma \frac{\partial L_{cla}^i}{\partial \theta_c}, \\ (\theta_d^j)^{i+1} &= (\theta_d^j)^i + \gamma\lambda \frac{\partial L_{con}^i(\theta_f, \theta_d^j)}{\partial \theta_d^j}, \\ \theta_p^{i+1} &= \theta_p^i + \gamma\lambda \frac{\partial L_{norm}^i}{\partial \theta_p}, \end{aligned} \quad (17)$$

where γ is the learning rate. It is very similar to the stochastic gradient descent (SGD). The only difference is in the updating of θ_p and θ_d^j, which contain the negative gradients from two domain classification losses. Such negative gradients contribute to making the learned features similar across domains. We propose a gradient-reversal layer (GRL) to update θ_f by easily following [21]. This gradient-reversal layer does nothing and merely forwards the input to the following layer during forward propagation. However, it multiplies the gradient by -1 during the backpropagation to obtain a negative gradient from the domain classification.

5 Experiments

In this section, we conduct experiments on three domain generalization datasets to demonstrate the effectiveness of our conditional invariant deep domain generalization (CIDDG). We compare our proposed method to the following methods.

- **L-SVM** [29] is support vector machines (SVM) with a linear kernel to classify the learned feature representations.
- **Kernel Fisher discriminant analysis (KDA)** [30] is used to find a transformation of data using nonlinear kernels in all source domains.
- **Undo-bias (UB)** [7] measures the model of each task with a domain-specific weight and a globally shared weight used for domain generalization. The original UB was developed for binary domain generalization classification. We extend it to a multi-class method using a one-vs-all strategy.
- **Domain-invariant component analysis (DICA)** [9] aims at learning a domain-invariant feature representation by matching the marginal distributions across domains.

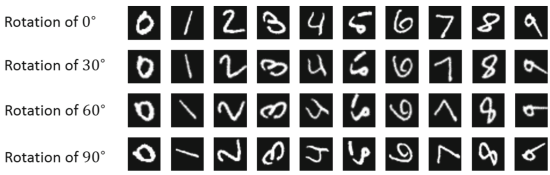


Fig. 2. Rotated MNIST dataset. Each rotation angle is viewed as one domain.

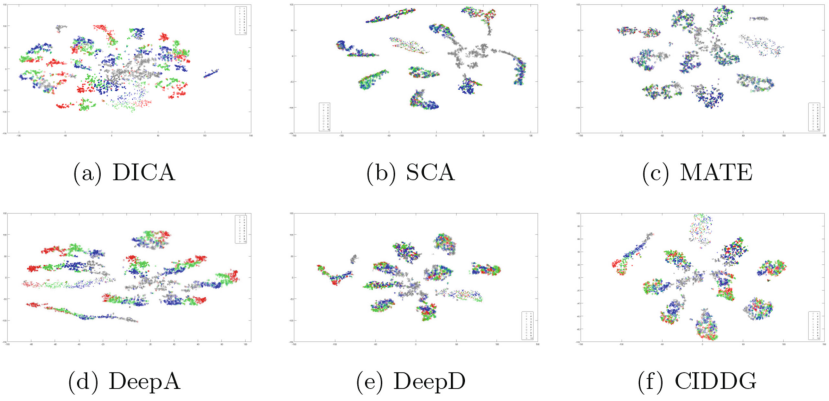
- **Scatter component analysis (SCA)** [8] is a unified framework for domain adaptation and domain generalization that also learns a domain-invariant feature transformation through marginal distributions.
- **Multi-task auto-encoder (MTAE)** [11] is a domain generalization method based on an auto-encoder to match marginal distributions across domains.
- **DeepA** refers to Deep-All, using data from all source domains to train the networks with only image classification loss.
- **DeepD** refers to Deep-Domain, using data from all source domains to train the networks with image classification loss and domain classification loss to match the marginal distribution $P(T(X))$.
- **DeepC** refers to Deep-Conditional, using data from all source domains to train networks with image classification loss and our proposed class-conditional domain classification loss in Eq. (13).
- **DeepN** refers to Deep-Normalize, using data from all source domains to train the networks with image classification loss and our proposed class prior-normalized domain classification loss in Eq. (14).
- **CIDDG** uses data from all source domains to train networks with image classification loss, class-conditional domain classification loss and class prior-normalized domain classification loss, as shown in Eq. (15).

5.1 Rotated MNIST Dataset

The rotated MNIST digits are shown in Fig. 2, which displays four different rotation angles: $0°, 30°, 60°$ and $90°$. Note that the original MNIST digits are already characterized by certain small-angle rotations. Each of the four rotation angles is viewed as one domain. Therefore, we have four domains. One domain is selected as the target domain and the other three ones are used as source domains. We repeat it four times, thus each domain is used as the target domain once. The number of training examples from each class in different domains are randomly chosen from a uniform distribution $U[80\ 160]$, to guarantee the variance of $P(Y)$ in each domain. The number of test examples is 10000 and they are obtained from the MNIST testset with corresponding rotation angles.

Table 1. Performance comparison in terms of accuracy (%) on rotated MNIST dataset.

Target	SVM	KDA	UB	DICA	SCA	MATE	DeepA	DeepD	DeepC	DeepN	CIDDG
Test0°	72.62	72.70	64.43	72.61	71.79	75.56	75.43	77.07	77.79	78.25	**78.47**
Test30°	92.17	91.95	89.60	90.72	91.85	92.84	93.44	94.16	94.11	94.71	**94.88**
Test60°	93.34	93.15	89.30	91.77	92.69	93.68	94.47	95.22	94.96	95.49	**95.64**
Test90°	77.62	72.81	69.39	72.05	73.43	78.34	79.56	82.95	80.08	83.99	**84.00**



(a) DICA (b) SCA (c) MATE

(d) DeepA (e) DeepD (f) CIDDG

Fig. 3. Feature visualization of different methods on rotated MNIST dataset when the target domain is 90°. Different colors refer to different domains and the gray color denotes the target domain. Different shapes corresponds to different image classes.

The network architecture for rotated MNIST is the same as the architecture in [31]. All domain classification networks consist of three fully-connected layers ($1024 \rightarrow 1024 \rightarrow 10$) and the GRL layer is connected to the ReLU layer after the last convolution layer. The input features for baseline methods (**SVM**, **KDA**,

UB, **DICA**, **SCA**, **MATE**) are extracted using the well-trained **DeepA** network. RBF kernel is applied to **KDA**, **UB**, **DICA** and **SCA**. Additionally, linear SVM is used to classify the learned domain-invariant features for **KDA**, **DICA**, **SCA** and **MATE**. Deep-learning-based methods, including **DeepA**, **DeepD**, **DeepC**, **DeepN** and **CIDDG**, use softmax layer to do the classification. The experimental results are summarized in Table 1.

Our proposed conditional-invariant adversarial network achieves the best performance when testing on different target domains. All deep-learning-based methods outperform traditional domain generalization methods. Our method can achieve better improvement on more challenging tasks, e.g. the target domain is 0° or 90°, which demonstrates that our method is more robust. When the target domain is 30° or 60°, the angle 30° or 60° can be interpolated from its corresponding source domains (0°, 60°, 90°) or (0°, 30°, 90°). It is easier to learn a generalized model when testing on an interpolation angle (30° or 60°) than testing on an extrapolation angle (0° or 90°).

To better understand the generalization ability of different methods, we also visualize the learned feature distribution using t-SNE [32] projection in Fig. 3. We randomly select 100 examples from each class in the target domain for visualization. In the visualization results, **DeepA** refers to original feature distribution learned by the network with just softmax loss. For **DeepA**, the feature has been learned to be discriminative but different domains are not well matched. Almost all domain generalization methods can learn better domain-invariant features. Methods like **SCA**, **MATE**, **DeepD** can well match the feature distributions of the source domains; however, the distributions of several classes in the target domain are not well matched. Note that the visualization performance of **KDA** are not promising, we do not show its visualization result considering the limited pages. For our **CIDDG**, the distributions of about two classes are not well matched. In genera, our **CIDDG** can learn more discriminative features, and better match the distributions across source domains and target domains.

5.2 VLCS Dataset

In this section, we conduct experiments on a real world image classification dataset VLCS. It consists of four different sub-datasets corresponding to four domains: PASCAL VOC2007 (V) [33], LabelMe (L) [34], Caltech-101 (C) [35] and SUN09 (S) [36]. Following the settings in previous works [7,9], we select the five shared classes (bird, car, chair, dog and person) for classification. The total image numbers in the four domains (V,L,C,S) are 3376, 2656, 1415 and 3282 respectively. We use AlexNet [1] to train all the deep learning models and extract the FC6 features as input for traditional baseline methods. All domain classification networks consist of 3 fully-connected layers ($1024 \rightarrow 1024 \rightarrow 3$) and the GRL layer is connected to the FC6 layer. The datasets from source domains are split into two parts: 70% for training and 30% for validation, following [11,28]. The whole target domain is used for testing.

For **SVM**, **KDA**, **UB**, **DICA**, **SCA**, **MATE**, we first directly extract FC6 features of AlexNet from source domains and then learn domain-invariant

features using these baseline domain generalization methods. Finally, linear SVM are used to train the classification model and test on target domains. For **DeepA**, we directly use all source domains to fine-tune the AlexNet and test on target domains. For **DeepD**, we use all the domains to fine-tune the AlexNet with a domain classification network to match the marginal distribution $P(X)$. **DeepC**, **DeepN** and **CIDDG** are our methods with different proposed losses. The experimental results are summarized in Table 2.

From the results, we can conclude that traditional domain generalization methods perform even worse than **DeepA** (the network just fine-tuned using all source domains). Deep domain generalization methods outperform the simply fine-tuned model **DeepA**. Additionally, our conditional-invariant domain generalization method performs better than domain generalization (**DeepD**) which matches the marginal distribution.

Table 2. Performance comparison in terms of accuracy (%) on VLCS dataset.

Target	SVM	KDA	UB	DICA	SCA	MATE	DeepA	DeepD	DeepC	DeepN	CIDDG
V	48.10	44.40	47.90	51.78	54.71	51.66	62.71	63.71	63.97	64.31	**64.38**
L	50.87	46.69	51.09	45.63	53.73	53.95	61.28	62.06	62.60	62.10	**63.06**
C	70.04	61.48	73.71	68.69	58.94	75.75	85.73	86.58	87.47	86.38	**88.83**
S	47.94	38.52	46.77	37.66	47.62	50.33	59.33	60.29	61.51	61.87	**62.10**

Table 3. Performance comparison in terms of accuracy (%) on PACS dataset.

Target	SVM	KDA	UB	DICA	SCA	MATE	DeepA	DeepD	DeepC	DeepN	CIDDG
P	55.15	59.04	55.57	55.93	59.10	58.44	77.98	80.39	**80.72**	77.45	78.65
A	41.80	47.66	42.48	47.46	50.05	45.95	57.55	60.75	62.30	59.17	**62.70**
C	52.30	53.29	48.93	57.00	58.79	51.11	67.04	68.63	69.58	67.86	**69.73**
S	47.87	48.21	46.30	40.70	50.62	49.25	58.52	60.76	64.45	60.92	**64.45**

5.3 PACS Dataset

PACS [28] consists of four sub-datasets corresponding to four different image styles, including photo (P), art-painting (A), cartoon (C) and sketch (S). Each image style can be viewed as one domain. The numbers of images in each domain are 1670, 2048, 2344, 3929 respectively. We use all the images from the source domains as train set and test on all the images from the target domain. We extract the features from FC7 layer for traditional methods and the GRL layer is also connected to FC7 layer. Other settings including the network architectures are the same as that used in VLCS dataset.

The results are shown in Table 3. Similar conclusions can be made as that in the experiments of VLCS dataset. The reason **DeepN** performs worse than **DeepC** is that PACS has larger data bias and variance of $P(Y)$. The class-conditional domain classification networks cannot learn well with just images in one specific image class and not considering the changes in $P(Y)$ across domains.

6 Conclusions

In this paper, we proposed a novel conditional-invariant deep domain generalization method. This method is superior than previous works because it matches conditional distributions by considering the changes in $P(Y)$ rather than marginal distributions, thus it can better learn domain invariant features. We prove that the distributions of multiple source domains can be perfectly matched with our proposed multi-player minimax value. Additionally, extensive experiments are conducted and the results demonstrate the effectiveness of our proposed method.

Acknowledgments. This work was supported by National Key Research and Development Program of China 2017YFB1002203, NSFC No. 61572451, and No. 61390514, Fok Ying Tung Education Foundation WF2100060004 and Youth Innovation Promotion Association CAS CX2100060016, and Australian Research Council Projects FL-170100117, DP-180103424, DP-140102164, LP-150100671.

References

1. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. Adv. Neural Inf. Process. Syst., 1097–1105 (2012)
2. 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)
3. Schroff, F., Kalenichenko, D., Philbin, J.: Facenet: a unified embedding for face recognition and clustering. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 815–823 (2015)
4. Parkhi, O.M., Vedaldi, A., Zisserman, A.: Deep face recognition. BMVC **1**, 6 (2015)
5. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: towards real-time object detection with region proposal networks. Adv. Neural Inf. Process. Syst., 91–99 (2015)
6. Torralba, A., Efros, A.A.: Unbiased look at dataset bias. In: 2011 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1521–1528. IEEE (2011)
7. Khosla, A., Zhou, T., Malisiewicz, T., Efros, A.A., Torralba, A.: Undoing the damage of dataset bias. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV 2012. LNCS, vol. 7572, pp. 158–171. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-33718-5_12
8. Ghifary, M., Balduzzi, D., Kleijn, W.B., Zhang, M.: Scatter component analysis: a unified framework for domain adaptation and domain generalization. IEEE Trans. Pattern Anal. Mach. Intell. **39**(7), 1414–1430 (2017)
9. Muandet, K., Balduzzi, D., Schölkopf, B.: Domain generalization via invariant feature representation. In: Proceedings of the 30th International Conference on Machine Learning (ICML 2013), pp. 10–18 (2013)
10. Xu, Z., Li, W., Niu, L., Xu, D.: Exploiting low-rank structure from latent domains for domain generalization. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8691, pp. 628–643. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10578-9_41

11. Ghifary, M., Bastiaan Kleijn, W., Zhang, M., Balduzzi, D.: Domain generalization for object recognition with multi-task autoencoders. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2551–2559 (2015)
12. Ding, Z., Fu, Y.: Deep domain generalization with structured low-rank constraint. IEEE Trans. Image Process. **27**(1), 304–313 (2018)
13. Zhang, K., Gong, M., Schölkopf, B.: Multi-source domain adaptation: a causal view. AAAI **I**, 3150–3157 (2015)
14. Li, Y., Gong, M., Tian, X., Liu, T., Tao, D.: Domain generalization via conditional invariant representations. AAAI **I**, 3579–3587 (2018)
15. Zhao, H., Zhang, S., Wu, G., Costeira, J.P., Moura, J.F., Gordon, G.J.: Multiple source domain adaptation with adversarial training of neural networks. arXiv preprint arXiv:1705.09684 (2017)
16. Liu, T., Tao, D., Song, M., Maybank, S.J.: Algorithm-dependent generalization bounds for multi-Task learning. IEEE Trans. Pattern Anal. Mach. Intell. **39**(2), 227–241 (2017)
17. Schölkopf, B., Janzing, D., Peters, J., Sgouritsa, E., Zhang, K., Mooij, J.: On causal and anticausal learning. arXiv preprint arXiv:1206.6471 (2012)
18. Janzing, D., Scholkopf, B.: Causal inference using the algorithmic markov condition. IEEE Trans. Inf. Theor. **56**(10), 5168–5194 (2010)
19. Lopez-Paz, D., Nishihara, R., Chintala, S., Schölkopf, B., Bottou, L.: Discovering causal signals in images. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2017) (2017)
20. Goodfellow, I. et al.: Generative adversarial nets. Adv. Neural Inf. Process. Syst., 2672–2680 (2014)
21. Ganin, Y., et al.: Domain-adversarial training of neural networks. J. Mach. Learn. Res. **17**(59), 1–35 (2016)
22. Long, M., Zhu, H., Wang, J., Jordan, M.I.: Deep transfer learning with joint adaptation networks. In: Precup, D., Teh, Y.W. (eds.) Proceedings of the 34th International Conference on Machine Learning. Volume 70 of Proceedings of Machine Learning Research., International Convention Centre, Sydney, Australia, PMLR, pp. 2208–2217, 6–11 Aug 2017
23. Zhang, K., Schölkopf, B., Muandet, K., Wang, Z.: Domain adaptation under target and conditional shift. In: International Conference on Machine Learning, pp. 819–827 (2013)
24. Gong, M., Zhang, K., Liu, T., Tao, D., Glymour, C., Schölkopf, B.: Domain adaptation with conditional transferable components. In: International Conference on Machine Learning, pp. 2839–2848 (2016)
25. Long, M., Wang, J., Ding, G., Sun, J., Yu, P.S.: Transfer feature learning with joint distribution adaptation. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2200–2207 (2013)
26. Courty, N., Flamary, R., Habrard, A., Rakotomamonjy, A.: Joint distribution optimal transportation for domain adaptation. arXiv preprint arXiv:1705.08848 (2017)
27. Lin, J.: Divergence measures based on the shannon entropy. IEEE Trans. Inf. Theor. **37**(1), 145–151 (1991)
28. Li, D., Yang, Y., Song, Y.Z., Hospedales, T.M.: Deeper, broader and artier domain generalization. In: 2017 IEEE International Conference on Computer Vision (ICCV), pp. 5543–5551. IEEE (2017)
29. Fan, R.E., Chang, K.W., Hsieh, C.J., Wang, X.R., Lin, C.J.: Liblinear: a library for large linear classification. J. Mach. Learn. Res. **9**, 1871–1874 (2008)

30. Mika, S., Ratsch, G., Weston, J., Scholkopf, B., Mullers, K.R.: Fisher discriminant analysis with kernels. In: Proceedings of the 1999 IEEE Signal Processing Society Workshop Neural Networks for Signal Processing IX, 1999, pp. 41–48. IEEE (1999)
31. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. **15**(1), 1929–1958 (2014)
32. Van Der Maaten, L.: Barnes-HUT-SNE. arXiv preprint arXiv:1301.3342 (2013)
33. Everingham, M., Van Gool, L., Williams, C.K., Winn, J., Zisserman, A.: The pascal visual object classes (VOC) challenge. Int. J. Comput. Vis. **88**(2), 303–338 (2010)
34. Russell, B.C., Torralba, A., Murphy, K.P., Freeman, W.T.: Labelme: a database and web-based tool for image annotation. Int. J. Comput. Vis. **77**(1), 157–173 (2008)
35. Griffin, G., Holub, A., Perona, P.: Caltech-256 object category dataset (2007)
36. Choi, M.J., Lim, J.J., Torralba, A., Willsky, A.S.: Exploiting hierarchical context on a large database of object categories. In: 2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 129–136. IEEE (2010)