International Journal of Machine Learning and Cybernetics (2023) 14:3457–3470
https://doi.org/10.1007/s13042-023-01843-4

ORIGINAL ARTICLE

Domain generalization by distribution estimation

Sentao Chen[1] · **Zijie Hong**[2]

Received: 17 August 2022 / Accepted: 16 April 2023 / Published online: 6 May 2023
© The Author(s), under exclusive licence to Springer-Verlag GmbH Germany, part of Springer Nature 2023

Abstract
Domain generalization generalizes a prediction model trained on multiple source domains to an unseen target domain. The source and target domains are different but related, making cross domain model generalization challenging but possible. Existing works assume that the domains are related by a feature transformation that makes the marginal distributions, the class-conditional distributions, or the posterior distributions similar among the domains, and learn this transformation via kernel mean matching or adversarial training. Here, in a neural network context we relate the source and target domains via the network mapping, innovatively learn this mapping by matching multiple source joint distributions to their mixture distribution, and simultaneously learn a subsequent probabilistic classifier for target domain classification. To quantify the discrepancy among the source joint distributions, we exploit the Kullback–Leibler (KL) divergence, and show that in our case the KL divergence can be approximated via estimating a domain label posterior distribution. We model this discrete posterior distribution as multiple linear functions, and obtain their optimal parameters in an analytic manner. The resulting cost function is a combination of the cross-entropy loss and the estimated KL divergence, which is directly minimized via optimizing the network parameters. The experiments on several publicly available datasets demonstrate the effectiveness of our proposal. We release the source code at https://github.com/sentaochen/Domain-Generalization-by-Distribution-Estimation.

Keywords Domain generalization · Distribution estimation · KL divergence · Neural network

1 Introduction

Supervised learning models (*e.g.*, Convolutional Neural Networks, CNNs) with appropriately learned parameters can generalize well to the test data [1, 2], under the assumption that both the training and test data are i.i.d. samples from a single joint probability distribution $P(\boldsymbol{x}, y)$ of the features $\boldsymbol{x}$ and the class label y. While this is a reasonable assumption to make, it could be violated in real-world applications. For example, in object recognition, the training and test visual data can be collected from different joint distributions, each of which represents a certain combination of image content, image style, illumination, and background [3, 4]. Under such circumstances, even the powerful and expressive CNN models may fail to produce accurate predictions for the test data.

Domain generalization [5, 6], the science of generalizing a prediction model across different domains, exactly aims at addressing this non-i.i.d. supervised learning problem. To be specific, its goal is to train a prediction model by leveraging labeled data from multiple source domains, and boost the generalization ability of this model in an unseen target domain. In line with the terminology in the work of Muandet et al. [6], here we refer to a domain as a joint probability distribution $P(\boldsymbol{x}, y)$. So far, domain generalization has been studied in various applications, such as action recognition [7, 8], object recognition [9, 10], and medical diagnosis [11, 12].

A basic premise behind domain generalization is that the unseen target domain should somehow relate to the source ones. Otherwise, it may be impractical to perform cross domain model generalization. From a statistical point of view, since a joint distribution $P(\boldsymbol{x}, y)$ can be decomposed into $P(\boldsymbol{x}, y) = P(\boldsymbol{x})P(y|\boldsymbol{x})$ or $P(\boldsymbol{x}, y) = P(\boldsymbol{x}|y)P(y)$, existing works generally assume that the source and target domains are related by a feature transformation, which

✉ Sentao Chen
sentaochenmail@gmail.com

Zijie Hong
thomas0@126.com

[1] Department of Computer Science, Shantou University, Daxue Road, Shantou 515063, Guangdong, China

[2] School of Software Engineering, South China University of Technology, Guangzhou Higher Education Mega Centre, Guangzhou 510006, Guangdong, China

makes the marginal distributions $P(\boldsymbol{x})$, the class-conditional distributions $P(\boldsymbol{x}|y)$, or the posterior distributions $P(y|\boldsymbol{x})$ similar among the domains [4, 6, 7, 13–15]. Based on this feature transformation and other assumptions on the distributions (*e.g.*, the prior distribution $P(y)$ being stable), the discrepancy between the source and target joint distributions is expected to decrease, and thus a source trained prediction model can generalize well to the target domain. Muandet et al. [6] assumed that the posterior distribution is stable across domains in the problem of automatic gating of flow cytometry data, and learned a dimension reduction matrix to match multiple source marginal distributions under the distributional variance, a metric that relies on the kernel mean embedding technique [16, 17]. Li et al. [7] continued to make this assumption in computer vision, and generalized a classification model to an unseen visual domain via learning domain invariant representations, which are produced by a neural network that matches the source marginal distributions in the activation space. However, from a causal learning perspective, reference [4] shows that for computer vision tasks such as object recognition, where the object classes are the causes of image features, not only the marginal distribution but also the posterior distribution can change across the visual domains. Accordingly, another line of works [3, 4, 18] propose to match the class-conditional distributions among multiple source domains, and assume that the prior distribution is stable. For instance, Conditional Invariant Deep Domain Generalization (CIDDG) [4] plays a minimax adversarial game between a network mapping and multiple discriminators to align the source class-conditional distributions in the activation space, which is also shown to be equivalent to matching these distributions under the generalized Jensen-Shannon (JS) divergence [19]. Due to a similar observation that the stability assumption of the posterior distribution may not hold, Zhao et al. [14] proposed to align the source posterior distributions as well as the source marginal distributions through adversarial training.

In this study, we address the domain generalization problem by exploiting the neural network model which contains a network mapping and a probabilistic classifier. In contrast to prior works [3, 4, 7, 14], we characterize the domain relationship as a network mapping that makes the source and target joint distributions similar. Under this characterization, we learn the network mapping via matching multiple source joint distributions to their mixture distribution, all of which are reflected by the accessible labeled source data, and learn a subsequent probabilistic model for target domain classification. After matching the distributions, we expect the network mapping to reduce the discrepancy between the source mixture joint distribution and the target joint distribution in the activation space, and consequently boost the generalization ability of the probabilistic classifier in the target domain.

To be specific, we match the source joint distributions under the Kullback-Leibler (KL) divergence, and show that by introducing a domain label variable l, the problem of approximating this divergence can be transformed into the problem of estimating a domain label posterior distribution $P(l|\boldsymbol{x}, y)$. We then model this discrete posterior distribution as multiple linear functions, and show that by minimizing the L^2-distance, the optimal parameters of these functions can be estimated analytically. As a result, we obtain an explicit estimate for the KL divergence. When matching distributions in the activation space, this explicit KL divergence estimate frees us from tackling the challenging minimax problems (*e.g.*, the ones in prior adversarial works [4, 14]), and allows us to solve a straightforward minimization problem similar to the ones in the kernel mean matching works [3, 6, 18]. Furthermore, to learn the downstream probabilistic classifier we minimize the typical cross-entropy loss. Our overall optimization model is a minimization problem that optimizes both the parameters of the network mapping and the probabilistic classifier to minimize a combination of the estimated KL divergence and the cross-entropy loss. In the remainder, we name our solution DGDE (Domain Generalization by Distribution Estimation) for convenience. To summarize, our contributions are as follows.

- We propose the DGDE solution for domain generalization, which trains a neural network by optimizing its parameters to minimize the estimated KL divergence among the source joint distributions in the activation space, and the cross-entropy loss of the probabilistic classifier.
- We show that in our case the KL divergence can be approximated via estimating a domain label posterior distribution, and that the distribution estimation can be conducted in an analytic manner by appropriately selecting the loss function and the hypothesis space. This brings an explicit estimate for the KL divergence, allowing us to solve a simple and straightforward minimization problem when matching the distributions.
- We demonstrate the effectiveness of DGDE on several real-world applications, including object recognition, action recognition, and face recognition.

2 Related work

We discuss the domain adaptation and domain generalization works that are most related to our solution.

2.1 Domain adaptation

Domain adaptation [20, 21] is closely related to domain generalization in the sense that it also aims at learning a prediction model from the source data and generalizing it to the related but differently distributed target data. The key difference between them is that in domain adaptation the unlabeled target data are available for training the prediction model, while in domain generalization they are only accessible when testing the model. In general, domain adaptation can be addressed by matching the distributions between domains and training a prediction model [22–26]. Cicek and Soatto [27] aligned the source and target class-conditional distributions, encouraged them to have disjoint support, and finally employed semi-supervised learning tools to improve the generalization ability of the classifier. Hu et al. [28] consistently aligned the marginal and class-conditional distributions between domains by constraining the gradient of marginal and class-conditional alignment to be synchronous. Yang et al. [29] proposed a bi-directional class-level adversaries framework for domain adaptation, which optimizes the bi-directional adversarial loss and the class-level discrepancy loss. Chen et al. [30] made use of the CNN to directly align the source and target joint distributions under the relative chi-squared divergence, and simultaneously learned a probabilistic classifier for classifying the target data.

Different from these domain adaptation methods, our domain generalization solution DGDE works under the setting where the unlabeled target data are not observed in advance, and improves the generalization ability of the neural network model in the unseen target domain by matching the source joint distributions.

2.2 Domain generalization

The problem of generalizing a prediction model from multiple source domains to an unseen target domain is first explored in machine learning and computer vision [5, 31]. Muandet et al. [6] formally introduced the terminology "domain generalization" for this problem, and improved a source trained classifier to an unseen target domain by proposing the Domain-Invariant Component Analysis (DICA) approach. In particular, DICA finds a feature transformation by minimizing the distributional variance among multiple source marginal distributions, and also preserves the functional relationship between input and output variables. Thereafter, matching the distributions of multiple source domains has become a fundamental solution to domain generalization [3, 4, 7, 14, 15, 32].

Scatter Component Analysis (SCA) [13] learns a projection matrix via minimizing the distributional variance among the source marginal distributions, as well as maximizing the separability of the classes and the separability of the unlabeled data. MMD-based Adversarial Auto-Encoder (MMD-AAE) [7] aligns the distributions of the coded source features via minimizing the Maximum Mean Discrepancy (MMD) [17], and simultaneously matches the aligned distribution to a prior Laplacian distribution via minimizing the chi-squared divergence between them. Based on a famous adversarial work [22], Adversarial Feature Learning with Accuracy Constraint (AFLAC) [32] not only learns source domain invariant features, but also ensures that the domain invariance does not interfere with the classification accuracy. By contrast, Conditional Invariant Domain Generalization (CIDG) [18] finds a dimension reduction matrix to match the source class-conditional distributions, and the source class prior-normalized marginal distributions, both under the distributional variance. This approach is then extended to its end-to-end deep counterpart CIDDG [4], in which the projection matrix is replaced by the neural network mapping, and the distributional variance by the generalized JS divergence. Moreover, Domain Generalization via Entropy Regularization (DGER) [14] aligns the source marginal distributions, and further matches the source posterior distributions via entropy regularization.

Apart from distribution matching, domain generalization is also addressed in other manners [9, 33, 34]. Li et al. [8] designed an episodic training procedure to train a deep network in a way that exposes it to the distribution shift that characterizes a novel domain at runtime. Dou et al. [11] proposed to enforce semantic features via global class alignment and local sample clustering, with losses explicitly derived in an episodic learning procedure. Zhang et al. [34] proposed a disentangled learning framework for domain generalization, which separates semantic and variation representations into different subspaces while enforcing invariance constraints. Gao et al. [35] performed meta-learning to find a reusable white-box loss function, which is solved using the Implicit Function Theorem (IFT) to obtain gradients of the target domain performance with respect to the source domain loss parameters.

Our DGDE explores the distribution matching solution to domain generalization, but it is pretty different from the previous attempts [4, 6, 13–15, 18] in this line. In particular, DGDE directly matches the source joint distributions for domain generalization, rather than respectively matching their components (the marginal distributions, the class-conditional distributions, *etc*.), which is practiced in [6, 14, 18]. Additionally, as a crucial building block of DGDE, the explicit KL divergence approximator (*i.e*., Eq. (17)), which is derived via innovatively estimating the domain label posterior distribution, enables our approach to match the distributions via solving a simple minimization problem, rather than the challenging minimax problems tackled in prior works [4, 14], which also leverage the KL divergence for distribution comparison.

Table 1 Symbols and their descriptions

Symbol	Description
$\boldsymbol{x}, y, l$	Features, class label, domain label
c, n	Number of classes/source domains
m_l, m	Number of samples in domain l/all domains
γ, λ	Tradeoff/regularization parameter
$\boldsymbol{P}, \boldsymbol{H}$	Symmetric matrices
$\boldsymbol{b}^1, \ldots, \boldsymbol{b}^n$	Column vectors
$\widehat{\boldsymbol{\alpha}}^1, \ldots, \widehat{\boldsymbol{\alpha}}^n$	Learned parameters

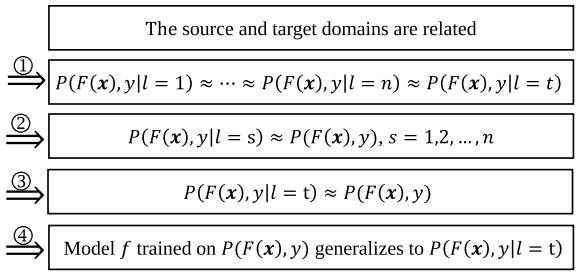

Fig. 1 The logic behind our solution to domain generalization

3 Domain generalization by distribution estimation

In this section, we first describe the domain generalization problem and present our motivation. Following that, we elaborate on the estimation of the distribution $P(l|\boldsymbol{x}, y)$ for divergence approximation in Sect. 3.1, and the estimation of the distribution $P(y|\boldsymbol{x})$ for classification in Sect. 3.2. Eventually, we present the optimization model and the learning algorithm in Sect. 3.3. For clarity and easy readability, we present in Table 1 an overview of the mathematical symbols used to describe our solution.

Let $\mathcal{X}$ be an input feature space, $\mathcal{Y} = \{1, \ldots, c\}$ be a class label space, and $\mathcal{L} = \{1, \ldots, n\}$ be a domain label space. With random variables $\boldsymbol{x} \in \mathcal{X}$, $y \in \mathcal{Y}$, and $l \in \mathcal{L}$, we define a joint probability distribution for each domain l as $P(\boldsymbol{x}, y|l)$. In domain generalization, the training data consist of n i.i.d. datasets $\mathcal{D}^1 = \{(\boldsymbol{x}_i^1, y_i^1)\}_{i=1}^{m_1}$, $\cdots$, $\mathcal{D}^n = \{(\boldsymbol{x}_i^n, y_i^n)\}_{i=1}^{m_n}$, which are respectively drawn from n related source domains $P(\boldsymbol{x}, y|l=1)$, $\cdots$, $P(\boldsymbol{x}, y|l=n)$. Note that, the union of these datasets can also be viewed as an i.i.d. set $\mathcal{D} = \mathcal{D}^1 \cup \ldots \cup \mathcal{D}^n = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{m}$ sampled from the source mixture joint distribution $P(\boldsymbol{x}, y) = \sum_{s=1}^{n} P(\boldsymbol{x}, y|l=s)P(l=s)$, where the number of samples $m = m_1 + \ldots + m_n$. Given these training data, the goal of domain generalization is to learn a classification model $f : \mathcal{X} \to \mathcal{Y}$ that generalizes well to an unknown but related target domain. Namely, the model should well predict the labels of samples governed by the target joint distribution $P(\boldsymbol{x}, y|l=t)$.

To ensure successful model generalization, it is crucial to exploit the relationship among domains. Here, we characterize the domain relationship as a neural network mapping F parameterized by $\boldsymbol{\Theta}_F$, which matches the source and target joint distributions in the activation space, *i.e.*, Fig. 1-①. Note that this characterization is appropriate and similar ones have also been introduced in [4, 14, 15]. Since both the target joint distribution and its random samples are not accessible, we therefore learn this mapping via matching the n source joint distributions to their mixture distribution in the activation space,[1] *i.e.*, Fig. 1-②. We expect that such a mapping F can also generalize to the target joint distribution and make it similar to the source mixture joint distribution, *i.e.*, Fig. 1-③. Under such circumstances, a source trained probabilistic classifier $f(\boldsymbol{x}) = \mathrm{argmax}_{y \in \mathcal{Y}} P(y|F(\boldsymbol{x};\boldsymbol{\Theta}_F);\boldsymbol{\Theta}_C)$ with $\boldsymbol{\Theta}_C$ being its parameter, generalizes well to the target domain, *i.e.*, Fig. 1-④. Specifically, we exploit the KL divergence and the cross-entropy loss to respectively quantify the distribution discrepancy and the classification loss, and match the distributions and learn the classification model via minimizing a cost function in the form

$$\mathcal{L}(\boldsymbol{\Theta}_C, \boldsymbol{\Theta}_F) = \mathcal{L}_{\mathrm{c}}(\boldsymbol{\Theta}_C, \boldsymbol{\Theta}_F) + \gamma \mathcal{L}_{\mathrm{d}}(\boldsymbol{\Theta}_F). \tag{1}$$

Here, $\mathcal{L}_{\mathrm{c}}(\boldsymbol{\Theta}_C, \boldsymbol{\Theta}_F)$ is the cross-entropy loss of the probabilistic classifier, $\mathcal{L}_{\mathrm{d}}(\boldsymbol{\Theta}_F)$ is the estimated KL divergence from the n source joint distributions to their mixture distribution in the activation space, and γ (> 0) is a tradeoff parameter for balancing the two terms. Below, we detail out the form of $\mathcal{L}_{\mathrm{d}}(\boldsymbol{\Theta}_F)$ and $\mathcal{L}_{\mathrm{c}}(\boldsymbol{\Theta}_C, \boldsymbol{\Theta}_F)$.

Remark 1 In real-world scenes, the objects from source domains $P(\boldsymbol{x}, y|l=1), \ldots, P(\boldsymbol{x}, y|l=n)$ may be quite different due to different camera viewpoints, backgrounds, or lighting conditions. By mapping the data to the activation space using the network mapping F, such redundant information irrelevant to object recognition could probably be reduced, making the source joint distributions similar in the space, *i.e.*, $P(F(\boldsymbol{x}), y|l=1) \approx \cdots \approx P(F(\boldsymbol{x}), y|l=n)$.

[1] Alternatively, one can also match the source joint distributions to each other, which will result in the heavy work of matching more pairs of distributions when the number of source domains $n \geq 4$.

3.1 Distribution estimation for divergence approximation

The KL divergence from the n source joint distributions $P(\boldsymbol{x}, y|l = 1), \ldots, P(\boldsymbol{x}, y|l = n)$ to their mixture distribution $P(\boldsymbol{x}, y) = \sum_{s=1}^{n} P(\boldsymbol{x}, y|l = s)P(l = s)$ is defined as[2]

$$\sum_{s=1}^{n} \mathrm{KL}\big(P(\boldsymbol{x}, y|l = s)\|P(\boldsymbol{x}, y)\big) = \sum_{s=1}^{n} \int P(\boldsymbol{x}, y|l = s) \log \frac{P(\boldsymbol{x}, y|l = s)}{P(\boldsymbol{x}, y)} d\boldsymbol{x}dy. \tag{2}$$

Clearly, Eq. (2) is non-negative and takes 0 when $P(\boldsymbol{x}, y|l = 1) = \cdots = P(\boldsymbol{x}, y|l = n) = P(\boldsymbol{x}, y)$.

In the following derivations, we show that the KL divergence in Eq. (2) can be expressed by the domain label posterior distribution $P(l|\boldsymbol{x}, y)$.

$$\sum_{s=1}^{n} \mathrm{KL}\big(P(\boldsymbol{x}, y|l = s)\|P(\boldsymbol{x}, y)\big) = \sum_{s=1}^{n} \int P(\boldsymbol{x}, y|l = s) \log \frac{P(l = s|\boldsymbol{x}, y)P(\boldsymbol{x}, y)}{P(\boldsymbol{x}, y)P(l = s)} d\boldsymbol{x}dy \tag{3}$$

$$= \sum_{s=1}^{n} \int P(\boldsymbol{x}, y|l = s) \log \frac{P(l = s|\boldsymbol{x}, y)}{P(l = s)} d\boldsymbol{x}dy \tag{4}$$

$$= \sum_{s=1}^{n} \int P(\boldsymbol{x}, y|l = s) \log P(l = s|\boldsymbol{x}, y) d\boldsymbol{x}dy - \sum_{s=1}^{n} \int P(\boldsymbol{x}, y|l = s) \log P(l = s) d\boldsymbol{x}dy \tag{5}$$

$$= \sum_{s=1}^{n} \int P(\boldsymbol{x}, y|l = s) \log P(l = s|\boldsymbol{x}, y) d\boldsymbol{x}dy - \sum_{s=1}^{n} \log P(l = s). \tag{6}$$

Equation (3) makes use of the Bayes' rule and writes $P(\boldsymbol{x}, y|l = s)$ as $P(\boldsymbol{x}, y|l = s) = \frac{P(l=s|\boldsymbol{x},y)P(\boldsymbol{x},y)}{P(l=s)}$ for $s \in \{1, 2, \ldots, n\}$. Equation (4) cancels out the factor $P(\boldsymbol{x}, y)$. Equation (5) expands the log term in Eq. (4). Equation (6) holds since $\int P(\boldsymbol{x}, y|l = s)d\boldsymbol{x}dy = 1$. For the first term in Eq. (6), we approximate the expectations with respect to distributions $P(\boldsymbol{x}, y|l = 1), \ldots, P(\boldsymbol{x}, y|l = n)$ by the empirical averages of their samples $\mathcal{D}^1, \ldots, \mathcal{D}^n$, and estimate the KL divergence as

$$\sum_{s=1}^{n} \mathrm{KL}\big(P(\boldsymbol{x}, y|l = s)\|P(\boldsymbol{x}, y)\big) \approx \sum_{s=1}^{n} \frac{1}{m_s} \sum_{i=1}^{m_s} \log P(l = s|\boldsymbol{x}_i^s, y_i^s) - \sum_{s=1}^{n} \log P(l = s), \tag{7}$$

where $(\boldsymbol{x}_i^s, y_i^s) \in \mathcal{D}^s$. As such, the problem of divergence approximation naturally transforms into the problem of estimating the domain label posterior distribution $P(l|\boldsymbol{x}, y)$.

To estimate the discrete posterior distribution $P(l|\boldsymbol{x}, y)$, we model it as multiple linear functions and learn the function parameters via minimizing the L^2-distance between distributions. As will be shown shortly, such a choice leads to the analytic solution of the parameters and consequently an explicit estimate of the KL divergence. Note that, with other choices like the nonlinear functions and other divergences (distances) between distributions, the analytic solution may not be possible. In particular, let us model the discrete domain label posterior distribution $P(l|\boldsymbol{x}, y)$ as:

$$P(l = 1|\boldsymbol{x}, y; \boldsymbol{\alpha}^1) = \sum_{i=1}^{m} \alpha_i^1 p\big((\boldsymbol{x}, y), (\boldsymbol{x}_i, y_i)\big), \tag{8}$$

$$\cdots,$$

$$P(l = n|\boldsymbol{x}, y; \boldsymbol{\alpha}^n) = \sum_{i=1}^{m} \alpha_i^n p\big((\boldsymbol{x}, y), (\boldsymbol{x}_i, y_i)\big), \tag{9}$$

where $p\big((\boldsymbol{x}, y), (\boldsymbol{x}_i, y_i)\big) = k(\boldsymbol{x}, \boldsymbol{x}_i)\delta(y, y_i)$ is a product of the feature and label kernels, and $\boldsymbol{\alpha}^s = (\alpha_1^s, \ldots, \alpha_m^s)^\top$ $(s = 1, \ldots, n)$ are the parameters of the functions. The feature kernel $k(\boldsymbol{x}, \boldsymbol{x}_i) = \exp\left(\frac{-\|\boldsymbol{x}-\boldsymbol{x}_i\|^2}{\sigma}\right)$ is the Gaussian kernel with positive kernel width σ, and the label kernel $\delta(y, y_i)$ is the delta kernel that evaluates 1 if $y = y_i$ and 0 otherwise. The linear-in-parameter functions from Eqs. (8) to (9) resemble the Radial Basis Function (RBF) networks and are reasonable choices for function approximation [36]. We learn parameters $\boldsymbol{\alpha}^s$ by matching $P(l = s|\boldsymbol{x}, y; \boldsymbol{\alpha}^s)$ to the true distribution $P(l = s|\boldsymbol{x}, y)$ under the L^2-distance:

$$(\boldsymbol{\alpha}_{\mathrm{opt}}^1, \ldots, \boldsymbol{\alpha}_{\mathrm{opt}}^n) = \underset{(\boldsymbol{\alpha}^1, \cdots, \boldsymbol{\alpha}^n)}{\mathrm{argmin}} \Big(\int \sum_{s=1}^{n} \big(P(l = s|\boldsymbol{x}, y) - P(l = s|\boldsymbol{x}, y; \boldsymbol{\alpha}^s)\big)^2 \times P(\boldsymbol{x}, y) d\boldsymbol{x}dy \Big) \tag{10}$$

[2] In an earlier work [19], this divergence is also called the generalized Jensen-Shannon divergence. There, the author defined it for measuring the discrepancy among multiple distributions.

$$= \underset{(\boldsymbol{\alpha}^1,\ldots,\boldsymbol{\alpha}^n)}{\operatorname{argmin}} \Big(\sum_{s=1}^{n} \int P(l=s|\boldsymbol{x},y;\boldsymbol{\alpha}^s)^2 P(\boldsymbol{x},y) d\boldsymbol{x}dy - 2 \int P(l|\boldsymbol{x},y;\boldsymbol{\alpha}^l) P(\boldsymbol{x},y,l) d\boldsymbol{x}dydl \Big). \tag{11}$$

Here, the objective function in Eq. (10) is the L^2-distance between posterior distributions $P(l=s|\boldsymbol{x},y)$ and $P(l=s|\boldsymbol{x},y;\boldsymbol{\alpha}^s)$. Equation (11) expands the quadratic term in Eq. (10) and discards the constant $\sum_{s=1}^{n} \int P(l=s|\boldsymbol{x},y)^2 P(\boldsymbol{x},y) d\boldsymbol{x}dy$. By approximating expectations via sample averages, we arrive at the empirical counterpart of Eq. (11):

$$(\widehat{\boldsymbol{\alpha}}^1,\ldots,\widehat{\boldsymbol{\alpha}}^n) = \underset{(\boldsymbol{\alpha}^1,\ldots,\boldsymbol{\alpha}^n)}{\operatorname{argmin}} \Big(\frac{1}{m} \sum_{s=1}^{n} \sum_{i=1}^{m} P(l=s|\boldsymbol{x}_i,y_i;\boldsymbol{\alpha}^s)^2 - \frac{2}{m} \sum_{i=1}^{m} P(l_i|\boldsymbol{x}_i,y_i;\boldsymbol{\alpha}^{l_i}) + \lambda \sum_{s=1}^{n} \|\boldsymbol{\alpha}^s\|^2 \Big) \tag{12}$$

$$= \underset{(\boldsymbol{\alpha}^1,\ldots,\boldsymbol{\alpha}^n)}{\operatorname{argmin}} \Big(\frac{1}{m} \sum_{s=1}^{n} (\boldsymbol{\alpha}^s)^\top (\boldsymbol{P}^\top \boldsymbol{P}) \boldsymbol{\alpha}^s - \frac{2}{m} \sum_{s=1}^{n} \mathbf{1}_{m_s}^\top \boldsymbol{P}^s \boldsymbol{\alpha}^s + \lambda \sum_{s=1}^{n} (\boldsymbol{\alpha}^s)^\top \boldsymbol{\alpha}^s \Big) \tag{13}$$

$$= \underset{(\boldsymbol{\alpha}^1,\ldots,\boldsymbol{\alpha}^n)}{\operatorname{argmin}} \sum_{s=1}^{n} \big((\boldsymbol{\alpha}^s)^\top (\boldsymbol{H} + \lambda \mathbf{I}_m) \boldsymbol{\alpha}^s - 2(\boldsymbol{b}^s)^\top \boldsymbol{\alpha}^s \big) \tag{14}$$

$$= \big((\boldsymbol{H} + \lambda \mathbf{I}_m)^{-1} \boldsymbol{b}^1, \ldots, (\boldsymbol{H} + \lambda \mathbf{I}_m)^{-1} \boldsymbol{b}^n \big). \tag{15}$$

In Eq. (12), a regularization term $\lambda \sum_{s=1}^{n} \|\boldsymbol{\alpha}^s\|^2$ with regularization parameter λ (> 0) is added to the empirical averages to avoid overfitting. Equation (13) writes the objective function in matrix form, where $\mathbf{1}_{m_s}$ is a m_s-dimensional column vector of ones, $\boldsymbol{P}^s \in \mathbb{R}^{m_s \times m}$, and $\boldsymbol{P} = \big((\boldsymbol{P}^1)^\top, \ldots, (\boldsymbol{P}^n)^\top \big) \in \mathbb{R}^{m \times m}$. The (i,j)-th element of $\boldsymbol{P}^s$ is defined as $p_{ij}^s = p\big((\boldsymbol{x}_i^s, y_i^s), (\boldsymbol{x}_j, y_j) \big)$. Equation (14) introduces three notations, where $\boldsymbol{H} = \frac{1}{m} \boldsymbol{P}^\top \boldsymbol{P}$, $\boldsymbol{b}^s = \frac{1}{m} (\boldsymbol{P}^s)^\top \mathbf{1}_{m_s}$, and $\mathbf{I}_m$ is the $m \times m$ identity matrix. These notations explicitly make Eq. (14) an unconstrained quadratic optimization problem, whose analytic solution is then presented in Eq. (15). Because a probability distribution is non-negative and sums up to 1, we process the estimated domain label posterior distribution as

$$P_{\text{nor}}(l=s|\boldsymbol{x},y;\widehat{\boldsymbol{\alpha}}^s) = \frac{\max\{10^{-8}, P(l=s|\boldsymbol{x},y;\widehat{\boldsymbol{\alpha}}^s)\}}{\sum_{j=1}^{n} \max\{10^{-8}, P(l=j|\boldsymbol{x},y;\widehat{\boldsymbol{\alpha}}^j)\}}. \tag{16}$$

Plugging this estimated distribution into Eq. (7), we obtain the KL divergence approximator:

$$\sum_{s=1}^{n} \widehat{\mathrm{KL}}\big(P(\boldsymbol{x},y|l=s) \| P(\boldsymbol{x},y) \big) = \sum_{s=1}^{n} \frac{1}{m_s} \sum_{i=1}^{m_s} \log P_{\text{nor}}(l=s|\boldsymbol{x}_i^s, y_i^s; \widehat{\boldsymbol{\alpha}}^s) - \sum_{s=1}^{n} \log P(l=s). \tag{17}$$

According to the above derivations, the estimated KL divergence from the n source joint distributions to their mixture distribution in the activation space, $\mathcal{L}_{\text{d}}(\boldsymbol{\Theta}_F)$, is therefore defined as

$$\mathcal{L}_{\text{d}}(\boldsymbol{\Theta}_F) = \sum_{s=1}^{n} \frac{1}{m_s} \sum_{i=1}^{m_s} \log P_{\text{nor}}(l=s|F(\boldsymbol{x}_i^s;\boldsymbol{\Theta}_F), y_i^s; \widehat{\boldsymbol{\alpha}}^s), \tag{18}$$

where $F(\boldsymbol{x};\boldsymbol{\Theta}_F)$ denotes the activation features produced by the network mapping F. Note that since our goal is to minimize the estimated divergence via optimizing the network mapping, we therefore drop the term $\sum_{s=1}^{n} \log P(l=s)$, which is clearly independent of the network mapping.

3.2 Distribution estimation for classification

After matching distributions via the network mapping F, we learn a downstream probabilistic model for target domain classification. To be specific, we aim to estimate another posterior distribution $P(y|F(\boldsymbol{x};\boldsymbol{\Theta}_F);\boldsymbol{\Theta}_C)$, which is the ultimate softmax output of the network. Following the common practice in [4, 32, 37], we exploit the cross-entropy loss to quantify the loss of this probabilistic model and define $\mathcal{L}_{\text{c}}(\boldsymbol{\Theta}_C, \boldsymbol{\Theta}_F)$ as

$$\mathcal{L}_{\text{c}}(\boldsymbol{\Theta}_C, \boldsymbol{\Theta}_F) = \frac{-1}{m} \sum_{i=1}^{m} \sum_{j=1}^{c} \delta(y_i, j) \log P(y=j|F(\boldsymbol{x}_i;\boldsymbol{\Theta}_F);\boldsymbol{\Theta}_C), \tag{19}$$

where $\delta(\cdot,\cdot)$ is the delta kernel function previously defined in Sect. 3.1.

3.3 Optimization model and learning algorithm

Algorithm 1 Learning Algorithm for DGDE

Input: Datasets:$\mathcal{D}^1, \cdots, \mathcal{D}^n$; Parameters:$\lambda, \gamma, \eta$.
Output: Optimized parameters $\{\mathbf{\Theta}_F^*, \mathbf{\Theta}_C^*\}$

```
Initialize {Θ_F, Θ_C}
repeat
    Sample a minibatch of data from each D^i
    Compute L_d(Θ_F) by Eq. (18)
    Compute L_c(Θ_C, Θ_F) by Eq. (19)
    L(Θ_C, Θ_F) = L_c(Θ_C, Θ_F) + γ L_d(Θ_F)
    Θ_F ← Θ_F − η ∇_{Θ_F} L(Θ_C, Θ_F)
    Θ_C ← Θ_C − η ∇_{Θ_C} L(Θ_C, Θ_F)
until maximum iteration reached
```

Putting Eqs. (18) and (19) together, we present the optimization model of our DGDE solution as

$$\min_{\mathbf{\Theta}_C, \mathbf{\Theta}_F} \mathcal{L}(\mathbf{\Theta}_C, \mathbf{\Theta}_F) = \mathcal{L}_{\mathrm{c}}(\mathbf{\Theta}_C, \mathbf{\Theta}_F) + \gamma \mathcal{L}_{\mathrm{d}}(\mathbf{\Theta}_F). \tag{20}$$

Note that, our solution is general and can be implemented with either shallow or deep neural network model. As aforementioned, in the network model, the network mapping is parameterized by $\mathbf{\Theta}_F$, and the downstream probabilistic classifier is parameterized by $\mathbf{\Theta}_C$. In the experiments, we implement our DGDE with both shallow and deep neural network models to show its effectiveness.

We employ the minibatch Stochastic Gradient Descent (SGD) algorithm to solve Problem (20), and provide the pseudo code of the optimization procedure in Algorithm 1. In the algorithm, $\nabla_{\mathbf{\Theta}_F} \mathcal{L}(\mathbf{\Theta}_C, \mathbf{\Theta}_F)$ is the gradient with respect to $\mathbf{\Theta}_F$, $\nabla_{\mathbf{\Theta}_C} \mathcal{L}(\mathbf{\Theta}_C, \mathbf{\Theta}_F)$ is the gradient with respect to $\mathbf{\Theta}_C$, and η (> 0) is the learning rate.

4 Experiments

Below, we evaluate our DGDE solution on 6 real-world visual datasets (see Fig. 2), which are popular in the domain generalization and domain adaptation literature [7, 14, 38, 39]. We start by describing the datasets in Sect. 4.1, then introduce the experimental setup in Sect. 4.2, present the experimental results in Sect. 4.3, and eventually finish by conducting the empirical analysis in Sect. 4.4. Our solution is implemented using Pytorch,[3] and the experiments are run on a PC equipped with a NVIDIA RTX 3090 GPU and 24 G RAM.

[3] https://pytorch.org/

4.1 Datasets

We first summarize the statistics of the datasets in Table 2, and then describe each dataset in the following.

IXMAS [40] is cross-view action recognition dataset. It contains videos of 11 human actions recorded from 5 different views (domains): View0 (**V0**), View1 (**V1**), View2 (**V2**), View3 (**V3**), and View4 (**V4**). Following prior works [7, 8, 13], we keep the first 5 actions and exclude the irregular actions, resulting in 91 image samples in each domain. See Fig. 2(a) for the example images.

Office-Caltech [41] contains 4 different visual object datasets: Amazon (**A**), Caltech (**C**), DSLR (**D**), and Webcam (**D**), which are acquired in different environments and share 10 object categories. In the experiments, each dataset is regarded as a domain and the number of samples in each domain is 958, 1123, 157, and 295, respectively. See Fig. 2(b) for the example images.

PIE-Multiview [39] is a face recognition dataset containing face images of 67 individuals captured from different views, illumination conditions, and expressions. This

Table 2 Statistics of the visual datasets

Dataset	Subset (Domain)	#Samples	#Classes
IXMAS	View0	91	5
	View1	91	5
	View2	91	5
	View3	91	5
	View4	91	5
Office-Caltech	Amazon	958	10
	Caltech	1123	10
	DSLR	157	10
	Webcam	295	10
PIE-Multiview	C27	1404	67
	C09	1407	67
	C05	1407	67
	C37	1404	67
	C25	1407	67
	C02	1407	67
VLCS	VOC2007	3376	5
	LabelMe	2656	5
	Caltech-101	1415	5
	SUN09	3282	5
PACS	ArtPainting	2048	7
	Cartoon	2344	7
	Photo	1670	7
	Sketch	3929	7
Office-Home	Art	2421	65
	Clipart	4379	65
	Product	4428	65
	RealWorld	4357	65

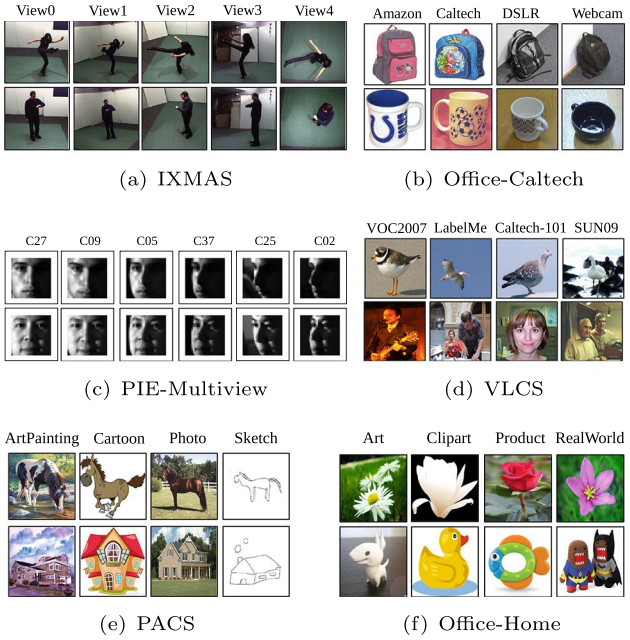

Fig. 2 Example images from 6 datasets. **a** IXMAS [40]. **b** Office-Caltech [41]. **c** PIE-Multiview [39]. **d** VLCS [42]. **e** PACS [43]. **f** Office-Home [44]

dataset has 6 subsets (domains): looking forward (**C27**), looking-downward (**C09**), looking towards left in an increasing angle (**C05**, **C37**, **C25**, **C02**). These 6 domains respectively contain 1404, 1407, 1407, 1404, 1407, and 1407 face images. See Fig. 2(c) for the example images.

VLCS [42] contains images from 4 well-known datasets (domains): VOC2007 (**V**) [45], LabelMe (**L**) [46], Caltech-101 (**C**) [47], and SUN09 (**S**) [48]. These domains (V, L, C, S) share 5 categories (*i.e.*, bird, car, chair, dog, and person) and have 3376, 2656, 1415, and 3282 image samples in each of them. See Fig. 2(d) for the example images.

PACS [43] is composed of 4 subsets corresponding to 4 different image styles: ArtPainting (**A**), Cartoon (**C**), Photo (**P**), and Sketch (**S**), with images from 7 classes: dog, elephant, giraffe, guitar, house, horse, and person. In the experiments, each image style is viewed as a domain, and the number of images in each domain is 2048, 2344, 1670, and 3929, respectively. See Fig. 2(e) for the example images.

Office-Home [44] is a large visual recognition dataset that comprises 4 domains: Art (**A**, artistic depictions of objects), Clipart (**C**, clipart images), Product (**P**, objects without a background) and RealWorld (**R**, objects captured with a regular camera). There are 65 categories shared by these domains, and the number of images in each domain is 2421, 4379, 4428, and 4357, respectively. See Fig. 2(f) for the example images.

4.2 Experimental setup

4.2.1 Comparison methods

Our DGDE solution is general and can be implemented with both shallow and deep neural networks. For completeness, we therefore respectively compare our shallow and deep implementations against existing shallow and deep domain generalization methods. To be specific, the shallow competitors include DICA [6], SCA [13], CIDG [18], and Multidomain Discriminant Analysis (MDA) [3]. Since these are dimensionality reduction methods, we enable their classification ability via appending a softmax classifier to them. The end-to-end deep competitors encompass Deeper, Broader and Artier Domain Generalization (DBADG) [43], CIDDG [4], Cross-Gradient (CrossGrad) [49], Jigsaw puzzle based Generalization (JiGen) [9], Deep Domain-Adversarial Image Generation (DDAIG) [50], Mixture of Multiple Latent Domains (MMLD) [51], EISNet [52], Representation Self-Challenging (RSC) [53], DGER [14], Domain Invariant Representation learning with domain Transformations via Generative Adversarial Networks (DIRT-GAN) [15], Adversarial Teacher-Student Representation Learning (ATSRL) [54], Stochastic Weight Averaging Densely (SWAD) [55], and Proxy-based Contrastive Learning (PCL) [56].

4.2.2 Evaluation protocol

We run the domain generalization methods on n source domains, *i.e.*, a collection of n image subsets here, and then employ them to classify the samples from an unseen target domain, *i.e.*, a held-out image subset. This is also known as the leave-one-domain-out evaluation protocol [4, 8, 11, 57]. Particularly, following [7, 39, 58], we run the shallow methods on the IXMAS dataset with the 5000-dimensional dense trajectories features [13], on the Office-Caltech dataset with the handcrafted 800-dimensional SURF features [41], and on the PIE-Multiview dataset with the 1024-dimensional gray-scale image pixel features [39]. Moreover, we run the deep end-to-end methods on the VLCS dataset with the AlexNet [1] backbone following [4, 14, 15, 43], and on the PACS and Office-Home datasets with the ResNet50 [2] backbone following [14, 53, 54, 56]. To ensure fair comparison with prior deep results, on the VLCS dataset, we randomly divide each domain into a training set (70%) and a test set (30%), and evaluate on the test set of the held-out target domain [14]. On the PACS and Office-Home datasets, we split the data from the source domains to 9 (train): 1 (val) and test on the whole held-out target domain [54]. Finally, along the general practice in domain generalization, the multi-class classification accuracy (%) on the target domain is adopted as the performance metric for all the methods.

Table 3 Classification accuracy (%) of shallow domain generalization methods on dataset IXMAS. Under the leave-one-domain-out evaluation protocol, the names of the source domains are omitted

Method	V0	V1	V2	V3	V4	Avg
DICA [6]	86.81	94.51	96.70	90.11	72.53	88.13
SCA [13]	89.01	97.80	96.70	92.31	69.23	89.01
CIDG [18]	92.61	**100.00**	96.70	**94.51**	70.33	90.83
MDA [3]	**93.41**	97.80	95.60	90.11	70.33	89.45
DGDE (ours)	**93.41**	**100.00**	**98.90**	**94.51**	**78.82**	93.13

In each column, the best result is highlighted in bold

Table 4 Classification accuracy (%) of shallow domain generalization methods on dataset Office-Caltech

Method	A	C	D	W	Avg
DICA [6]	52.51	46.48	66.36	60.34	56.42
SCA [13]	52.40	46.39	63.06	55.93	54.45
CIDG [18]	54.28	46.48	65.35	59.64	56.44
MDA [3]	53.55	47.20	66.76	58.98	56.62
DGDE (ours)	**54.91**	**47.91**	**71.97**	**66.44**	**60.31**

In each column, the best result is highlighted in bold

4.2.3 Implementation details

On datasets IXMAS, Office-Caltech, and PIE-Multiview, we implement our DGDE solution using the one-Hidden-Layer Neural Networks (1HLNN) with 2500, 400, and 512 hidden neurons, respectively. Thus, on each dataset the number of hidden neurons for each network is half of the number of its input neurons. These shallow networks with the ReLU activation are trained from scratch by the minibatch SGD with the learning rate η set to 10^{-3} on IXMAS, and to 10^{-2} on Office-Caltech and PIE-Multiview. Besides, the tradeoff parameter γ is selected from the range $\{10^{-2}, 10^{-1}, 1, 10^{1}, 10^{2}\}$ via cross-validation on the training data. On datasets VLCS, PACS, and Office-Home, we start training our solution with the CNN backbones from their ImageNet pre-trained models following [14, 56]. The optimizer is still the minibatch SGD with the learning rate $\eta = 10^{-3}$. Furthermore, this time γ is not selected through a grid search as in the experiments with shallow networks, since the corresponding procedure would be computationally costly. Instead, following [22], γ is initiated at 0 and is gradually changed to 1 using the formula $\gamma = \frac{2}{1+\exp(-10\times \text{iter}/\text{total\_iter})} - 1$, where iter is the current iteration times and total_iter is the total iteration times. For all the experiments, a minibatch in every iteration of the SGD algorithm consists of n minibatches, which are randomly sampled from the n source domains. The Gaussian kernel width σ is set to the median squared distance between the training data in a manner similar to [59]. The regularization parameter λ is fixed at 10^{-2}, whose sensitivity analysis will be presented in Sect. 4.4.4. Additionally, following [11, 14, 57] we repeat the experiment on each task with different random seeds, and report the average classification accuracy of our approach over 5 independent runs.

4.3 Experimental results

4.3.1 Results of shallow domain generalization methods

We report the experimental results of the shallow domain generalization methods on datasets IXMAS, Office-Caltech, and PIE-Multiview in Tables 3, 4, and 5, respectively. In every table, the names of the source domains are omitted under the leave-one-domain-out evaluation protocol. For every column in the table, the best result is highlighted in **bold**.

The results from Tables 3 to 5 evidence that on majority of the tasks, our DGDE solution with shallow implementation performs significantly better than the comparison methods. In particular, DICA and SCA do not perform well since they only match the marginal distributions and neglect matching the posterior distributions, which is critical in domain generalization [14, 15]. By contrast, our DGDE solution aims at matching the joint distributions, which includes matching the marginal distributions and the

Table 5 Classification accuracy (%) of shallow domain generalization methods on dataset PIE-Multiview

Method	C27	C09	C05	C37	C25	C02	Avg
DICA [6]	90.88	73.70	88.98	76.99	53.38	58.55	73.75
SCA [13]	93.60	75.42	88.63	78.78	53.52	57.00	74.49
CIDG [18]	98.43	**83.87**	94.74	82.34	60.48	65.81	80.95
MDA [3]	97.72	81.31	94.24	80.48	57.64	58.92	78.39
DGDE (ours)	**99.15**	82.02	**96.59**	**90.74**	**67.02**	**73.21**	**84.79**

In each column, the best result is highlighted in bold

Table 6 Classification accuracy (%) of deep domain generalization methods with the AlexNet backbone on dataset VLCS. Under the leave-one-domain-out evaluation protocol, the names of the source domains are omitted

Method	V	L	C	S	Avg
DBADG [43]	69.99	63.49	93.64	61.32	72.11
CIDDG [4]	73.00	58.30	97.02	68.89	74.30
JiGen [9]	70.62	60.90	96.93	64.30	73.19
MMLD [51]	71.96	58.77	96.66	68.13	73.88
DGER [14]	73.24	58.26	96.92	69.10	74.38
DIRT-GAN [15]	72.10	64.00	97.30	**72.20**	76.40
DGDE (ours)	**74.27**	**64.15**	**98.35**	70.25	**76.76**

In each column, the best result is highlighted in bold

Table 7 Classification accuracy (%) of deep domain generalization methods with the ResNet50 backbone on dataset PACS

Method	A	C	P	S	Avg
EISNet [52]	86.64	81.53	97.11	78.07	85.84
DDAIG [50]	85.40	78.50	95.70	80.00	84.90
RSC [53]	87.89	82.16	97.92	83.35	87.83
DGER [14]	87.51	79.31	98.25	76.30	85.34
ATSRL [54]	90.00	83.50	**98.90**	80.00	88.10
SWAD [55]	89.30	83.40	97.30	82.50	88.10
PCL [56]	90.20	83.90	98.10	82.60	88.70
DGDE (ours)	**90.41**	**84.20**	98.86	**83.56**	**89.26**

In each column, the best result is highlighted in bold

posterior distributions. Therefore, our solution naturally outperforms its competitors. Note that to construct the domain generalization tasks from PIE-Multiview, *i.e.*, Table 5, we follow the popular leave-one-domain-out evaluation protocol [4, 8, 11, 57]. However, on this dataset with 6 domains, we are curious about how our solution will behave under another protocol of leaving several domains out. Hence, we will explore this later in Sect. 4.4.2.

Table 8 Classification accuracy (%) of deep domain generalization methods with the ResNet50 backbone on dataset Office-Home

Method	A	C	P	R	Avg
CrossGrad [49]	67.70	57.70	79.10	80.40	71.23
DDAIG [50]	65.20	59.20	77.70	76.70	69.70
ATSRL [54]	69.30	60.10	**81.50**	82.10	73.25
SWAD [55]	66.10	57.70	78.40	80.20	70.60
PCL [56]	67.30	59.90	78.70	80.70	71.60
DGDE (ours)	**70.55**	**60.76**	80.05	**82.30**	**73.42**

In each column, the best result is highlighted in bold

4.3.2 Results of deep domain generalization methods

We report the experimental results of the deep domain generalization methods on datasets VLCS, PACS, and Office-Home in Tables 6, 7, and 8, respectively. Note that, the results of the deep comparison methods are all cited from prior works, since the corresponding experimental settings are the same. To be specific, the results of the comparison methods in Table 6 are quoted from [14, 15, 60], the results in Table 7 from [14, 54, 56], and the results in Table 8 from [54, 56].

From the results in Tables 6, 7, and 8, we observe that on most tasks, our solution with deep implementation again outperforms its comparison methods, some of which are complex methods with a large set of model parameters and involve the heavy work of tuning quite a few parameters (*e.g.*, MASF, DGER). For example, in Table 7, our DGDE performs much better than MASF and DGER on every task, and also yields superior results to the most recent method PCL. We believe that in domain generalization where one does not have much knowledge about the target domain, the solution should aim at addressing the most critical problem. Our DGDE is exactly targeted at tackling the critical joint distribution mismatch problem in domain generalization [14, 15], and therefore produces better results than the other methods. Together with the previous shallow results,

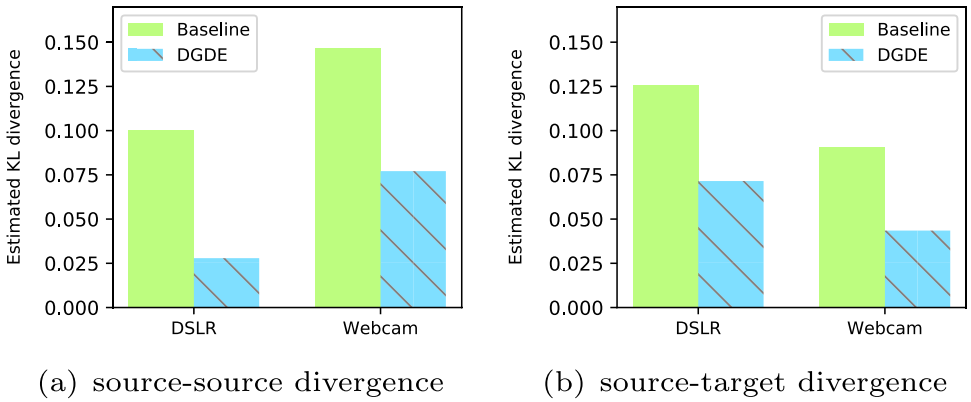

Fig. 3 Estimated KL divergence values computed using the activation features from the Baseline and our DGDE networks, which are trained on the tasks with the target domains being DSLR and Webcam. **a** Divergence values from the source joint distributions to their mixture distribution. **b** Divergence values from the source mixture joint distribution to the target joint distribution

Table 9 Classification accuracy (%) of shallow domain generalization methods on the PIE-Multiview dataset under the leave-several-domains-out evaluation protocol

Source	Target	DICA [6]	SCA [13]	CIDG [18]	MDA [3]	DGDE (ours)
C09, C05, C25	C27	90.53	90.53	98.22	97.15	**98.65**
	C37	68.80	69.23	73.15	71.30	**80.06**
	C02	48.97	48.90	51.88	48.19	**60.55**
	Avg	69.43	69.55	74.42	72.21	**79.75**

In each column, the best result is highlighted in bold

we remark that our solution of matching the source joint distributions under our KL divergence approximator is indeed effective for addressing the domain generalization problem. And our solution works well with both the shallow models and the end-to-end deep architectures.

4.4 Empirical analysis

4.4.1 Model assumption

We examine the model assumption behind our solution to better understand why it can improve the generalization ability of a neural network in the domain generalization problem. To this end, we first check the estimated KL divergence from the source joint distributions to their mixture distribution, using the activation features from the Baseline and our DGDE networks. Here, the Baseline trains a normal neural network by simply minimizing the cross-entropy loss, *i.e.*, the one defined in Eq. (19). The Baseline and DGDE networks are trained on the domain generalization tasks from dataset Office-Caltech, with the target domains being DSLR and Webcam. The results are illustrated in Fig. 3(a). We observe from this figure that the two divergence values of DGDE are much smaller than the ones of the Baseline, indicating that our DGDE network mappings (*i.e.*, the hidden layers) are indeed effective in reducing the discrepancy among multiple source joint distributions in the activation space. More importantly, we then check the estimated KL divergence from the source mixture joint distribution to the target joint distribution, and depict the results in Fig. 3(b). From Fig. 3(b), we again observe that the two divergence values of DGDE are below the ones of the Baseline. This suggests that while the differently distributed target data are not available for training our DGDE networks, on the tested evaluations the mappings of our DGDE networks successfully generalize to the related target joint distributions, and narrow down the distribution gap between source and target. Consequently, our DGDE networks perform well in the target domain classification tasks.

4.4.2 Leave several domains out

We study our solution under the interesting evaluation protocol of leave-several-domains-out. Namely, we first run our DGDE on the same set of source domains, and then observe how the resulting network model will generalize to several different but related target domains. In particular, we employ the PIE-Multiview dataset with 6 domains, and construct 3 domain generalization tasks with the same set of source domains {C09, C05, C25} and the different target domains C27, C37, and C02. Note that such construction ensures that the 3 different target domains are all related to the source ones and hence the possibility of cross domain model generalization. Subsequently, we run our DGDE and other shallow domain generalization methods on {C09, C05, C25}, and report in Table 9 the classification accuracy in target domains C27, C37, and C02. Clearly, the results in Table 9 show that our DGDE network (a single network) manages to generalize well to different target domains and consistently outperforms the other domain generalization methods. This confirms that our solution is also effective in this new evaluation protocol for domain generalization.

Table 10 Comparison of the methods in terms of their parameters (> 0)

Method	DICA [6]	SCA [13]	CIDG [18]	MDA [3]	DGDE (ours)
Parameter	λ, ϵ	β, δ	α, γ	λ, β, γ	λ, γ

Table 11 Comparison of the methods in terms of their execution time (second)

Method	DICA [6]	SCA [13]	CIDG [18]	MDA [3]	DGDE (ours)
Time	6.0	8.0	13.0	16.0	22.0

4.4.3 Further comparison

We compare our solution with other methods in terms of their parameters and present the results in Table 10. We note that our solution has the same number of parameters as most of the comparison methods. Specifically, as aforementioned, we set the value of our parameter γ by cross-validation or by following the strategy in [22], and fix the other

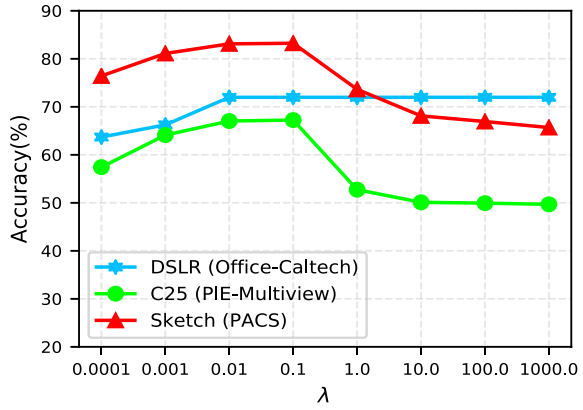

Fig. 4 Parameter sensitivity of DGDE on domain generalization tasks from different datasets

parameter $\lambda = 10^{-2}$, whose sensitivity analysis is presented in Sect. 4.4.4.

We compare our solution with others in terms of their execution time (second). Table 11 reports the comparison results on the domain generalization task where the target domain is Amazon (Office-Caltech). The results are recorded under the same environment. According to Table 11, we note that while our DGDE solution is not computationally more efficient than the comparison methods, its execution time is within a reasonable range, considering its superior performance.

4.4.4 Parameter sensitivity

We investigate the sensitivity of DGDE with respect to different choices of its parameter λ, which is kept fixed at 10^{-2} in the main experiments. To this end, we run the sensitivity experiments on the domain generalization tasks where the target domains are DSLR (Office-Caltech), C25 (PIE-Multiview), and Sketch (PACS), with λ varying in the range $\{10^{-4}, 10^{-3}, \cdots, 10^{3}\}$. Figure 4 depicts the classification accuracy of DGDE versus different choices of λ on these tasks. From Fig. 4, we observe that on different tasks from different datasets, DGDE attains its superior performance when λ is around 10^{-2}. Therefore, as a general guideline for choosing this parameter, we would suggest fixing $\lambda = 10^{-2}$.

4.4.5 Feature visualization

We exploit the t-SNE visualization tool [61] and visualize in Fig. 5(a)-Fig. 5(d) the source and target data from the activation spaces of DDAIG, SWAD, PCL, and DGDE. The domain generalization task has the source domains Clipart,

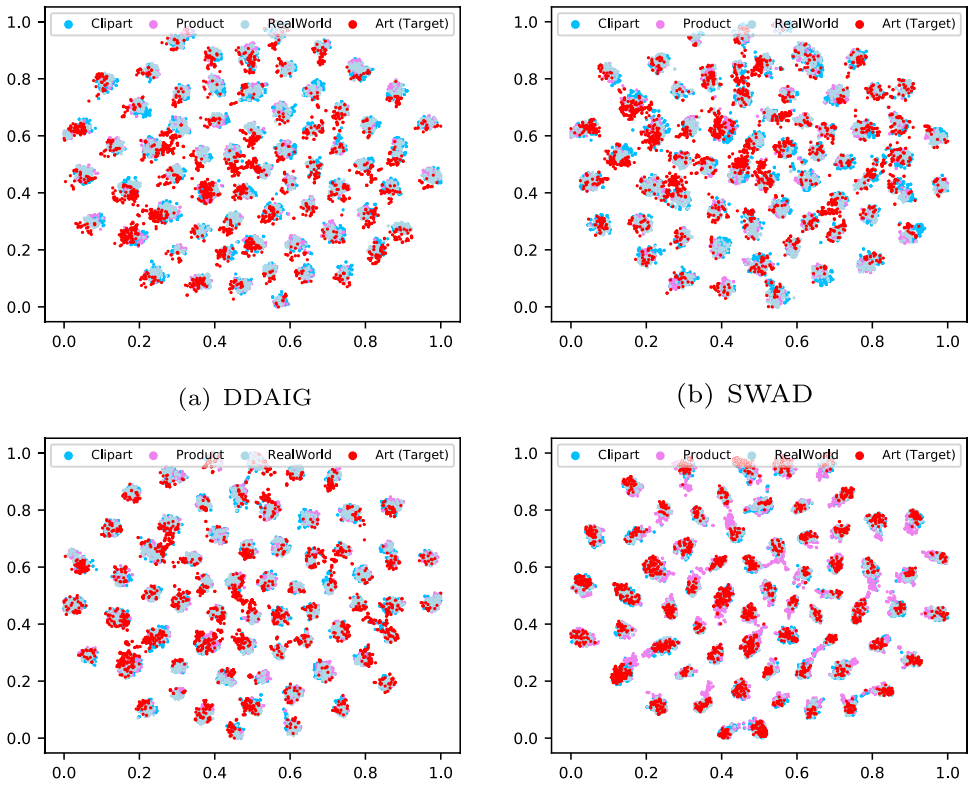

Fig. 5 T-SNE visualization of source and target data in the activation spaces of DDAIG, SWAD, PCL, and DGDE. The source domains are Clipart, Product, and RealWorld, and the target domain is Art. **a** DDAIG, **b** SWAD, **c** PCL, **d** DGDE

Product, and RealWorld, and the target domain Art from the Office-Home dataset. By comparing Fig. 5(d) against Fig. 5(a), Fig. 5(b), and Fig. 5(c), we observe that our DGDE solution better aligns the source and target data in the network activation space than its competitors DDAIG, SWAD, and PCL. These visualization results show that our joint distribution matching under the KL divergence is a powerful solution to domain generalization.

5 Conclusion

In this paper, we propose the DGDE solution to domain generalization, which minimizes the discrepancy among the source joint distributions to match them in the neural network activation space, and optimizes the probabilistic classifier for target domain classification. Specifically, we quantify the distribution discrepancy using the KL divergence, and innovatively derive the KL approximator via estimating the domain label posterior distribution. This distribution is estimated using multiple linear-in-parameter functions and the L^2-distance, leading to the analytic solution of the parameters. In the experiments, we implement the proposed DGDE solution with both shallow and deep neural network models, and show the power of these implementations on several publicly available datasets. As a future work, we intend to explore the application of our DGDE solution to the problem of RGB-Infrared cross-modality person ReID (RGB-IR ReID) [62], where the RGB images and infrared (IR) images come from different but related modalities.

Acknowledgements This work was supported in part by the National Natural Science Foundation of China under Grant 62106137, in part by the Guangdong Basic and Applied Basic Research Foundation under Grant 2023A1515012954, and in part by Shantou University under Grant NTF21035.

Data availibility Datasets are open source and can be obtained through website links or references.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Krizhevsky A, Sutskever I, Hinton GE. Imagenet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems, pp. 1097–1105 (2012)
2. He K, Zhang X, Ren S, Sun J. Deep residual learning for image recognition. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
3. Hu S, Zhang K, Chen Z, Chan L. Domain generalization via multidomain discriminant analysis. In: Conference on Uncertainty in Artificial Intelligence, vol. 35 (2019)
4. Li Y, Tian X, Gong M, Liu Y, Liu T, Zhang K, Tao D. Deep domain generalization via conditional invariant adversarial networks. In: European Conference on Computer Vision, pp. 624–639 (2018)
5. Blanchard G, Lee G, Scott C. Generalizing from several related classification tasks to a new unlabeled sample. In: Advances in Neural Information Processing Systems, pp. 2178–2186 (2011)
6. Muandet, K., Balduzzi, D., Schölkopf, B.: Domain generalization via invariant feature representation. In: International Conference on Machine Learning, vol. 28, pp. 10–18 (2013)
7. Li H, Pan SJ, Wang S, Kot AC. Domain generalization with adversarial feature learning. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 5400–5409 (2018)
8. Li D, Zhang J, Yang Y, Liu C, Song Y-Z, Hospedales TM. Episodic training for domain generalization. In: IEEE International Conference on Computer Vision, pp. 1446–1455 (2019)
9. Carlucci FM, D'Innocente A, Bucci S, Caputo B, Tommasi T. Domain generalization by solving jigsaw puzzles. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 2224–2233 (2019)
10. Chen S, Wang L, Hong Z, Yang X (2023) Domain generalization by joint-product distribution alignment. Pattern Recogn 134:109086
11. Dou, Q., de Castro, D.C., Kamnitsas, K., Glocker, B.: Domain generalization via model-agnostic learning of semantic features. In: Advances in Neural Information Processing Systems, pp. 6450–6461 (2019)
12. Blanchard G, Deshmukh AA, Dogan U, Lee G, Scott C (2021) Domain generalization by marginal transfer learning. J Mach Learn Res 22(2):1–55
13. Ghifary M, Balduzzi D, Kleijn WB, Zhang M (2017) Scatter component analysis: A unified framework for domain adaptation and domain generalization. IEEE Trans Pattern Anal Mach Intell 39(7):1414–1430
14. Zhao, S., Gong, M., Liu, T., Fu, H., Tao, D.: Domain generalization via entropy regularization. In: Advances in Neural Information Processing Systems, vol. 33, pp. 3118–3129 (2020)
15. Nguyen AT, Tran T, Gal Y, Baydin AG. Domain invariant representation learning with domain density transformations. In: Advances in Neural Information Processing Systems, vol. 34, pp. 5264–5275 (2021)
16. Sriperumbudur BK, Gretton A, Fukumizu K, Schölkopf B, Lanckriet GR (2010) Hilbert space embeddings and metrics on probability measures. J Mach Learn Res 11:1517–1561
17. Gretton A, Borgwardt KM, Rasch MJ, Schölkopf B, Smola A (2012) A kernel two-sample test. J Mach Learn Res 13:723–773
18. Li Y, Gong M, Tian X, Liu T, Tao D. Domain generalization via conditional invariant representations. In: AAAI Conference on Artificial Intelligence, pp. 3579–3587 (2018)
19. Lin J (1991) Divergence measures based on the shannon entropy. IEEE Trans Inform Theory 37(1):145–151
20. Jiang J. A literature survey on domain adaptation of statistical classifiers. URL: http://sifaka. cs. uiuc. edu/jiang4/domainadaptation/survey 3, 1–12 (2008)
21. Kouw WM, Loog M (2021) A review of domain adaptation without target labels. IEEE Trans Pattern Anal Mach Intell 43(3):766–785
22. Ganin Y, Ustinova E, Ajakan H, Germain P, Larochelle H, Laviolette F, March M, Lempitsky V (2016) Domain-adversarial training of neural networks. J Mach Learn Res 17(59):1–35
23. Chen S, Yang X (2019) Tailoring density ratio weight for covariate shift adaptation. Neurocomputing 333:135–144
24. Chen S, Han L, Liu X, He Z, Yang X (2020) Subspace distribution adaptation frameworks for domain adaptation. IEEE Trans Neural Netw Learn Syst 31(12):5204–5218

25. Noori Saray S, Tahmoresnezhad J (2022) Iterative joint classifier and domain adaptation for visual transfer learning. Int J Mach Learn Cybern 13(4):947–961
26. Chen Q, Zhang H, Ye Q, Zhang Z, Yang W (2022) Learning discriminative feature via a generic auxiliary distribution for unsupervised domain adaptation. Int J Mach Learn Cybern 13(1):175–185
27. Cicek S, Soatto S. Unsupervised domain adaptation via regularized conditional alignment. In: IEEE International Conference on Computer Vision, pp. 1416–1425 (2019)
28. Hu L, Kan M, Shan S, Chen X. Unsupervised domain adaptation with hierarchical gradient synchronization. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 4043–4052 (2020)
29. Yang G, Ding M, Zhang Y (2022) Bi-directional class-wise adversaries for unsupervised domain adaptation. Appl Intell 52(4):3623–3639
30. Chen S, Hong Z, Harandi M, Yang X. Domain neural adaptation. IEEE Transactions on Neural Networks and Learning Systems, 1–12 (2022)
31. Khosla A, Zhou T, Malisiewicz T, Efros AA, Torralba A. Undoing the damage of dataset bias. In: European Conference on Computer Vision, pp. 158–171 (2012). Springer
32. Akuzawa, K., Iwasawa, Y., Matsuo, Y.: Adversarial invariant feature learning with accuracy constraint for domain generalization. In: Joint European Conference on Machine Learning and Knowledge Discovery in Databases, pp. 315–331 (2019)
33. Chu X, Jin Y, Zhu W, Wang Y, Wang X, Zhang S, Mei H. DNA: Domain generalization with diversified neural averaging. In: International Conference on Machine Learning, pp. 4010–4034 (2022)
34. Zhang H, Zhang Y-F, Liu W, Weller A, Schölkopf B, Xing EP. Towards principled disentanglement for domain generalization. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 8024–8034 (2022)
35. Gao B, Gouk H, Yang Y, Hospedales T. Loss function learning for domain generalization by implicit gradient. In: International Conference on Machine Learning, pp. 7002–7016 (2022)
36. Que Q, Belkin M (2020) Back to the future: Radial basis function network revisited. IEEE Trans Pattern Anal Mach Intell 42(8):1856–1867
37. Chen S, Zheng L, Wu H. Riemannian representation learning for multi-source domain adaptation. Pattern Recognition, 109271 (2023)
38. Chen S, Harandi M, Jin X, Yang X (2021) Semi-supervised domain adaptation via asymmetric joint distribution matching. IEEE Trans Neural Netw Learn Syst 32(12):5708–5722
39. Herath S, Harandi M, Porikli F. Learning an invariant hilbert space for domain adaptation. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 3956–3965 (2017)
40. Weinland D, Ronfard R, Boyer E (2006) Free viewpoint action recognition using motion history volumes. Comput Vis Image Understand 104(2–3):249–257
41. Gong B, Shi Y, Sha F, Grauman K. Geodesic flow kernel for unsupervised domain adaptation. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 2066–2073 (2012)
42. Fang C, Xu Y, Rockmore DN. Unbiased metric learning: On the utilization of multiple datasets and web images for softening bias. In: IEEE International Conference on Computer Vision, pp. 1657–1664 (2013)
43. Li D, Yang Y, Song Y-Z, Hospedales TM. Deeper, broader and artier domain generalization. In: IEEE International Conference on Computer Vision, pp. 5542–5550 (2017)
44. Venkateswara H, Eusebio J, Chakraborty S, Panchanathan S. Deep hashing network for unsupervised domain adaptation. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 5018–5027 (2017)
45. Everingham M, Van Gool L, Williams CK, Winn J, Zisserman A (2010) The pascal visual object classes (voc) challenge. Int J Comput Vis 88(2):303–338
46. Russell BC, Torralba A, Murphy KP, Freeman WT (2008) Labelme: a database and web-based tool for image annotation. Int J Comput Vis 77(1–3):157–173
47. Fei-Fei L, Fergus R, Perona P. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. In: IEEE Conference on Computer Vision and Pattern Recognition Workshop, pp. 178–178 (2004). IEEE
48. Choi MJ, Lim JJ, Torralba A, Willsky AS. Exploiting hierarchical context on a large database of object categories. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 129–136 (2010). IEEE
49. Shankar, S., Piratla, V., Chakrabarti, S., Chaudhuri, S., Jyothi, P., Sarawagi, S.: Generalizing across domains via cross-gradient training. In: International Conference on Learning Representations (2018)
50. Zhou K, Yang Y, Hospedales T, Xiang T. Deep domain-adversarial image generation for domain generalisation. In: AAAI Conference on Artificial Intelligence, vol. 34, pp. 13025–13032 (2020)
51. Matsuura T, Harada T. Domain generalization using a mixture of multiple latent domains. In: AAAI Conference on Artificial Intelligence, vol. 34, pp. 11749–11756 (2020)
52. Wang S, Yu L, Li C, Fu C-W, Heng P-A. Learning from extrinsic and intrinsic supervisions for domain generalization. In: European Conference on Computer Vision, pp. 159–176 (2020)
53. Huang Z, Wang H, Xing EP, Huang D. Self-challenging improves cross-domain generalization. In: European Conference on Computer Vision, pp. 124–140 (2020)
54. Yang F-E, Cheng Y-C, Shiau Z-Y, Wang Y-CF. Adversarial teacher-student representation learning for domain generalization. In: Advances in Neural Information Processing Systems, vol. 34, pp. 19448–19460 (2021)
55. Cha J, Chun S, Lee K, Cho H-C, Park S, Lee Y, Park S. Swad: Domain generalization by seeking flat minima. In: Advances in Neural Information Processing Systems, vol. 34, pp. 22405–22418 (2021)
56. Yao X, Bai Y, Zhang X, Zhang Y, Sun Q, Chen R, Li R, Yu B. Pcl: Proxy-based contrastive learning for domain generalization. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 7097–7107 (2022)
57. Zhou K, Yang Y, Hospedales T, Xiang T. Learning to generate novel domains for domain generalization. In: European Conference on Computer Vision, pp. 561–578 (2020)
58. Ding Z, Fu Y (2018) Deep domain generalization with structured low-rank constraint. IEEE Trans Image Process 27(1):304–313
59. Baktashmotlagh M, Harandi M, Salzmann M (2016) Distribution-matching embedding for visual domain adaptation. J Mach Learn Res 17(108):1–30
60. Mansilla L, Echeveste R, Milone DH, Ferrante E. Domain generalization via gradient surgery. In: IEEE International Conference on Computer Vision, pp. 6630–6638 (2021)
61. Maaten LVD, Hinton G (2008) Visualizing data using t-sne. J Mach Learn Res 9:2579–2605
62. Wan L, Sun Z, Jing Q, Chen Y, Lu L, Li Z (2023) G2da: Geometry-guided dual-alignment learning for rgb-infrared person re-identification. Pattern Recogn 135:109150

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.