JUSTC

Article

http://justc.ustc.edu.cn

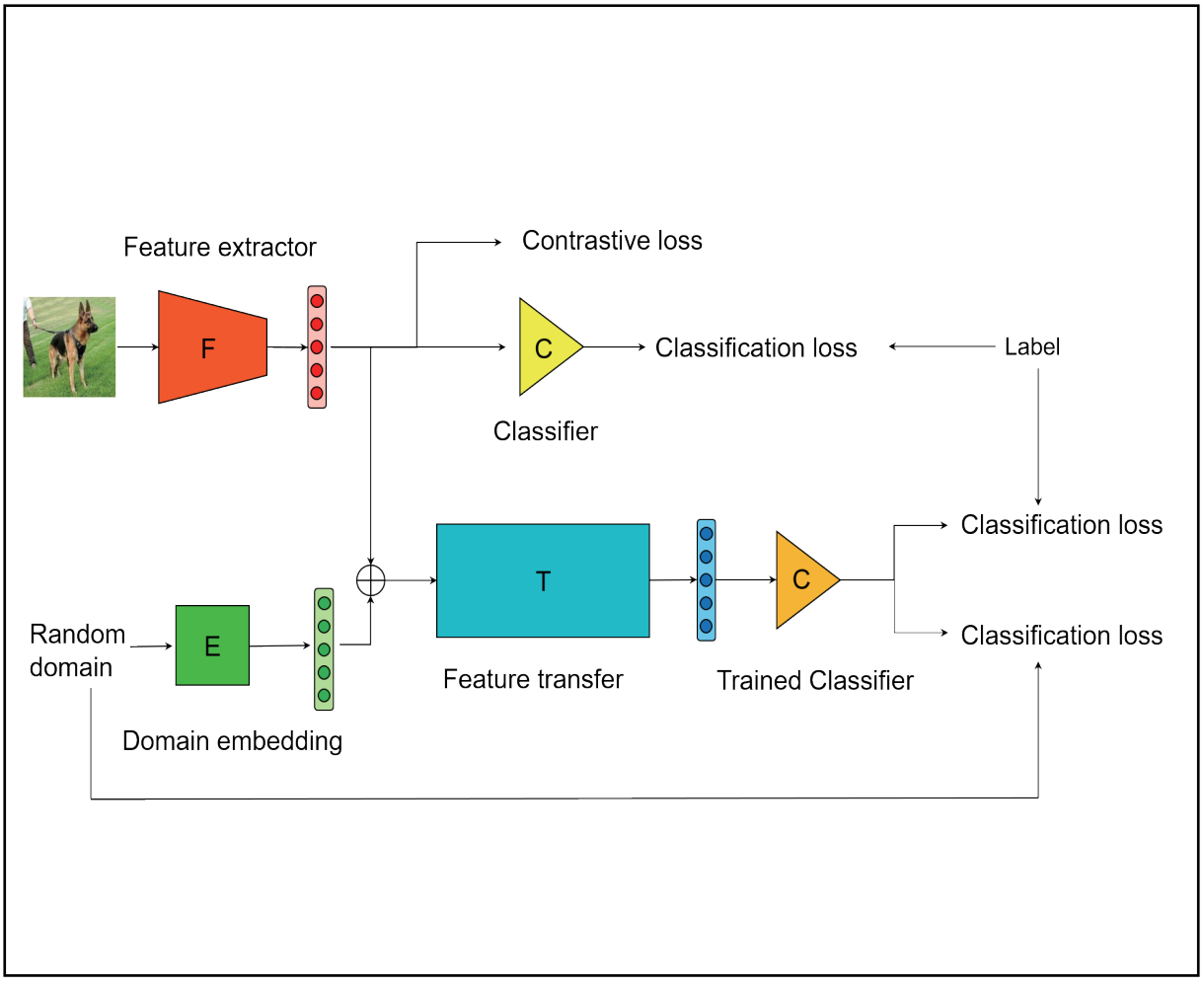
Received: January 20, 2023; Accepted: May 04, 2023

A feature-transfer-model based domain generalization method with a mixed contrastive loss

Yuesong Wang, and Hong Zhang

Department of Statistics and Finance, School of Management, University of Science and Technology of China, Hefei 230026, China

Graphical abstract



Public summary

- We propose a feature transfer model for domain generalization and a new sampling strategy based on Mixup in cooperation with contrastive loss.
- Experiments on two mainstream datasets demonstrate the superiority of our method.

Citation: Wang Y S, Zhang H. A feature-transfer-model based domain generalization method with a mixed contrastive loss. *JUSTC*, **2024**, 54(4): 0404. DOI: 10.52396/JUSTC-2023-0010

JUSTC

Article

http://justc.ustc.edu.cn

Received: January 20, 2023; Accepted: May 04, 2023

A feature-transfer-model based domain generalization method with a mixed contrastive loss

Yuesong Wang, and Hong Zhang

Department of Statistics and Finance, School of Management, University of Science and Technology of China, Hefei 230026, China

Cite This: *JUSTC*, **2024**, 54(4): 0404 (8pp)

Read Online

SI Supporting Information

Abstract: When domains, which represent underlying data distributions, differ between training and test datasets, traditional deep neural networks suffer from a substantial drop in their performance. Domain generalization methods aim to boost generalizability on an unseen target domain by using only training data from source domains. Mainstream domain generalization algorithms usually make modifications on some popular feature extraction networks such as ResNet, or add more complex parameter modules after the feature extraction networks. Popular feature extraction networks are usually well pre-trained on large-scale datasets, so they have strong feature extraction abilities, while modifications can weaken such abilities. Adding more complex parameter modules results in a deeper network and is much more computationally demanding. In this paper, we propose a novel feature transfer model based on popular feature extraction networks in domain generalization, without making any changes or adding any module. The generalizability of this feature transfer model is boosted by incorporating a contrastive loss and a data augmentation strategy (i.e., Mixup), and a new sample selection strategy is proposed to coordinate Mixup and contrastive loss. Experiments on the benchmarks PACS and Domainnet demonstrate the superiority of our proposed method against conventional domain generalization methods.

Keywords: Contrastive loss; data augmentation; deep neural network; domain generalization; feature transfer

CLC number: **Document code:** A

1 Introduction

It is critical for machine learning algorithms to maintain safe and reliable predictions that generalize well across domains. Machine learning models are usually applied in scenarios where the test data distribution is different from that of training data. This phenomenon has serious consequences, especially when the predictions are used for life-threatening events such as medical diagnosis[1-3]. The distributional gap between the training and test datasets raises the possibility that the prediction is significantly mistaken. A domain generalization algorithm aims to train a model on multiple source domains, which has sound generalization on unseen target domains. To accomplish this goal, models must be trained to capture useful features observed commonly in source domains.

Most domain generalization methods assume that different domains share some "stable" features whose relationship with the output is invariant across domains, and the goal of most domain generalization methods is to learn such domain-invariant features. Domain generalization methods can be grouped into several categories based on their techniques. Some algorithms define novel loss functions to learn domain-agnostic representations[4,5], which are applicable to a broad range of tasks. Alternatively, in recent years, researchers have been increasingly interested in designing deep neural network (DNN) architectures to achieve similar goals[6–9]. These DNN architectures usually achieve better performances on experimental datasets, but they usually modify popular feature extraction networks, such as ResNet, or add more complex parameter modules after the popular feature extraction network. Popular feature extraction networks are usually well pretrained on large-scale datasets so that they have strong feature extraction abilities, and modifications could weaken such abilities. Furthermore, adding more complex parameter modules results in a deeper network and is much more computationally demanding.

We aim to improve the widely used feature extraction networks from three aspects so that their outputs have a higher prediction generalizability. On the output side, a suitable loss function can be used to maximize the similarity of features from the same label, so we adopt a contrastive loss and redefine positive and negative samples in accordance with the domain generalization problem. On the input side, specific sampling strategies and data augmentation methods can be used to improve the generalization ability of the original networks, so we implement a new Mixup[10]-based sampling strategy. On the outside of the whole network, we design a feature transfer model for another prediction using random domain input and feature transfer, which can independently evaluate the generalization of output features based on a causal model.

The novelties of this paper are threefold. First, our method does not change the general feature extraction network and does not add any modules, so it can be easily extended to any other network. Second, we propose a new sample selection

DOI: 10.52396/JUSTC-2023-0010

JUSTC, **2024**, 54(4): 0404

strategy to coordinate Mixup and contrastive loss, which effectively improves the performance of our method. Third, our method significantly outperforms existing methods in applications to two benchmark datasets.

2 Related Work

2.1 Domain generalization

Algorithms for domain generalization usually require a variety of labeled training source domains. However, target data with a domain different from those of training data are not available during training for domain generalization methods[11,12]. Many early domain generalization methods[4,6,13] borrowed the idea of distribution alignment from domain adaption[14–17] to reduce the distributional gap between multiple training sources. Some recent domain generalization methods consider generating extra synthetic images given the multiple source domains so that test data whose distribution is usually close to that of the training data are "in-distribution" with the training data[18]. Some methods decompose network parameters into domain-specific and domain-invariant parts during training, while only domain-invariant parameters are used for predictions in test stage[19,20]. For instance, Peng et al.[21] developed a low-rank parameterized convolutional neural network (CNN) model where each layer of the network is decomposed into "common" and "specific" components. Several normalization and meta-learning strategies are also considered for domain generalization[22–24].

2.2 Contrastive loss

Contrastive loss is the crucial loss function in contrastive learning[25–30] which has superior performance in the context of self-supervised learning with applications to pretrain tasks and unsupervised learning problems. Most existing works on contrastive learning have focused on unsupervised algorithms, while this paper addresses domain generalization problems with the intent of discovering feature invariants. Mitrovic et al.[27] proposed a contrastive learning technique by imposing a specific prediction regularizer across augmentations to enhance in-class consistency. We extend this contrastive loss to our proposed method, albeit to an entirely different interpretation and application.

2.3 Mixup

Mixup[10] is a simple learning technique used to reduce unwanted oscillations due to some behaviors such as memorization and sensitivity arising from adversarial samples. Mixup trains a neural network on convex combinations of instance-label pairs through a regularization that favors straightforward linear behavior in-between training samples. Furthermore, Mixup can lessen the need to remember imperfect labels and improve the robustness with respect to adversarial samples.

3 Materials and methods

3.1 Problem formulation

The "Domain" could be considered as a particular type of data distribution. The key of popular models is the conditional distribution $P(Y|X)$, where X stands for the input and Y stands for the desired outcome. In the presence of multiple domains, the conditional distribution usually depends on domains and can be denoted by $P_d(Y|X)$, where d stands for the domain.

For domain generalization problems, we consider D source domains $\mathcal{D} = \{\mathcal{D}_d\}_{d=1}^{D}$, where the dth domain $\mathcal{D}_d$ consists of N_d training pairs $\{(x_{d,i}, y_{d,i})\}_{i=1}^{N_d}$, with $x_{d,i}$ being the ith input in $\mathcal{D}_d$ and $y_{d,i} \in \{1, 2, \ldots, n_c\}$ being the label of $x_{d,i}$. Here n_c is the total number of classes. The goal of the domain generalization method is to learn a model from multiple labeled source domains that generalize well to an unseen target domain D_T. In this paper, we focus on image classification tasks.

Our method is graphically illustrated in Fig. 1. In our method, a contrastive loss is helpful in extracting invariant information from images with the aid of a novel Mixup strategy. A feature transfer model is designed to improve and verify the

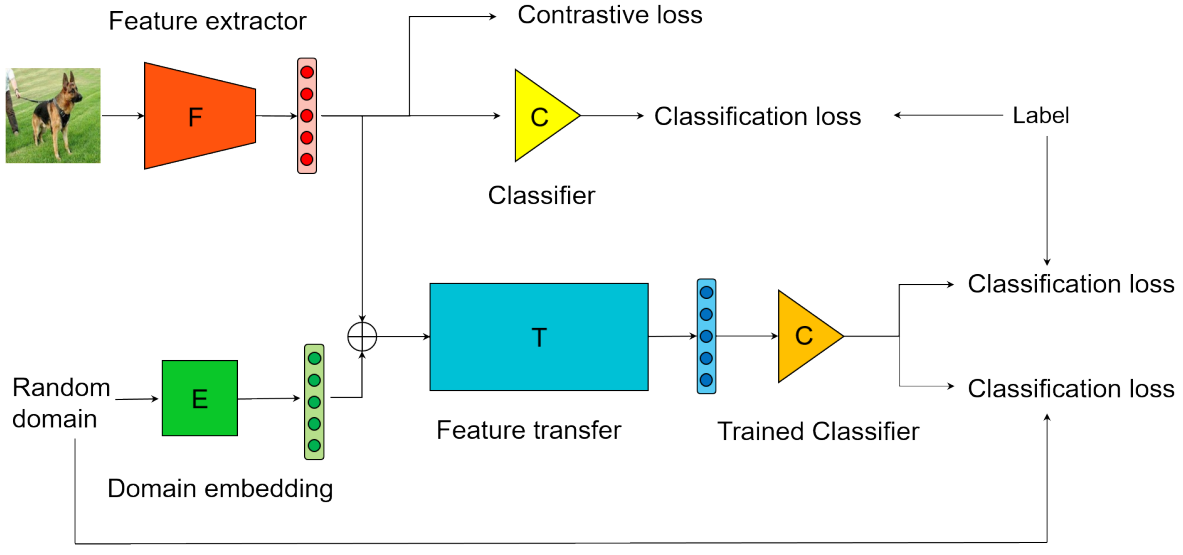


Fig. 1. Illustration of our proposed method.

DOI: 10.52396/JUSTC-2023-0010
JUSTC, **2024**, 54(4): 0404

generalizability of retrieved features.

Now we describe our method in detail. In the first stage, we encode each training sample using a feature-extracting encoder F (CNN in this paper). Specifically, we first apply the feature extractor F to transform an image x into a latent vector $f(x)$. In the second stage, we utilize the feature $f(x)$ to calculate four losses (on two sides), whose linear combination is the final optimization objective for model training. On one side, the feature $f(x)$ is sent to a contrastive loss L_{con}, and it is also sent to a label classifier to obtain a label classification loss L_{cls1} defined as

$$L_{\text{cls1}} = -\sum_{i=1}^{n_c} y^i \log(\hat{y}^i), \tag{1}$$

where n_c is the total number of classes, $(y_1,\cdots,y_{n_c})$ is the one-hot vector of label y with $y^i = 0$ for $i \neq y$ and $y^i = 1$ for $i = y$, and $\hat{y}^i$ is the output probability for the ith class by the classifier. On the other side, $f(x)$ is first combined with a domain feature $e(d)$ encoded from a random input domain d through a domain embedding encoder E. Then, the combined feature $(f(x), e(d))$ is sent to a feature transfer model T to obtain a new feature $t(x,d)$. Next, to identify label and domain, the new feature $t(x,d)$ is input into a trained classifier, producing a label classification loss L_{cls2} and a domain classification loss L_{cls3}. The specific formulations of L_{cls2} and L_{cls3} will be described in Section 3.4.

3.2 Contrastive loss

We aim to generate domain-invariant features. Such features should be invariant to domains and have high similarity scores for those samples with the same label. To this end, we adopt a contrastive loss function with inputs including positive and negative samples in addition to the original inputs, where positive and negative samples are generated as follows. For any sample x, we draw m_1 positive samples with labels the same as that of x and domains randomly selected from domains other than that of x. Similarly, m_2 negative samples have domains the same as x and labels randomly selected from labels other than that of x.

The contrastive loss is described as follows. Let $\{x_1^+,\cdots,x_{m_1}^+\}$ and $\{x_1^-,\cdots,x_{m_2}^-\}$ be positive samples and negative samples, respectively, which are drawn from the training data and are correlated with the original inputs x's. The contrastive loss is defined as

$$L_{\text{con}} = -\log\left(\frac{\sum_{i=1}^{m_1} \mathrm{e}^{f(x)^{\mathrm{T}} f(x_i^+)/\sqrt{h}}}{\sum_{i=1}^{m_1} \mathrm{e}^{f(x)^{\mathrm{T}} f(x_i^+)/\sqrt{h}} + \sum_{i=1}^{m_2} \mathrm{e}^{f(x)^{\mathrm{T}} f(x_i^-)/\sqrt{h}}}\right), \tag{2}$$

where h is the dimension of $f(x)$.

3.3 A Mixup based sampling strategy

Mixup is a commonly used technique for data augmentation by constructing virtual training samples as follows:

$$\begin{cases} \tilde{x}_0 = \lambda x_1 + (1-\lambda)x_2, \\ \tilde{y}_0 = \lambda y_1 + (1-\lambda)y_2, \end{cases} \tag{3}$$

where (x_1,y_1) and (x_2,y_2) are two samples (first original sample and second original sample hereafter) drawn randomly from our training data, with x_1 and x_2 being raw input vectors and y_1 and y_2 being one-hot label encodings, and $\lambda \in [0,1]$ is used to weight the two samples. With the virtual training samples, we need to generate their corresponding positive and negative samples.

Denote the domains of x_1 and x_2 by d_1 and d_2, respectively, the label and domain of any positive sample by y_p and d_p, respectively, and the label and domain of any negative sample by y_n and d_n, respectively. In what follows, we present our sampling strategies under five different circumstances in what follows (refer to Table 1 for a summary), where a domain/label is randomly selected from a candidate domain/label set for each circumstance.

(S1) If we do not use Mixup, then we choose those samples with the same label $y_p = y_1$ but a different domain $d_p \neq d_1$ as positive samples and those samples with a different label $y_n \neq y_1$ and the same domain $d_n = d_1$ as negative samples.

(S2) If we choose a sample with the same label $y_2 = y_1$ and the same domain $d_2 = d_1$ as the second original sample, then we choose two samples (labels $y_{p_1} = y_{p_2} = y_1$; domains $d_{p_1} = d_{p_2} \neq d_1$) to mix up a positive sample and two samples (labels $y_{n_1} = y_{n_2} \neq y_1$; domains $d_{n_1} = d_{n_2} = d_1$) to mix up a negative sample.

(S3) If we choose a sample with different label $y_2 \neq y_1$ and the same domain $d_2 = d_1$ as the second original sample, then we choose two samples (labels $(y_{p_1},y_{p_2}) = (y_1,y_2)$ or (y_2,y_1); domains $d_{p_1} = d_{p_2} \neq d_0$) to mix up a positive sample and two samples (labels $y_{n_1},y_{n_2} \neq y_1,y_2$; domains $d_{n_1} = d_{n_2} = d_1$) to mix up a negative sample.

Table 1. A brief illustration of five sampling strategies

	S1	S2	S3	S4	S5
y_2	None	$y_2 = y_1$	$y_2 \neq y_1$	$y_2 = y_1$	$y_2 \neq y_1$
d_2	None	$d_2 = d_1$	$d_2 = d_1$	$d_2 \neq d_1$	$d_2 \neq d_1$
y_p	$y_p = y_1$	$y_{p_1} = y_{p_2} = y_1$	$(y_{p_1},y_{p_2}) = (y_1,y_2)$ or (y_2,y_1)	$y_{p_1} = y_{p_2} = y_1$	$(y_{p_1},y_{p_2}) = (y_1,y_2)$ or (y_2,y_1)
d_p	$d_p \neq d_1$	$d_{p_1} = d_{p_2} \neq d_1$	$d_{p_1} = d_{p_2} \neq d_1$	$d_{p_1},d_{p_2} \neq d_1,d_2$	$d_{p_1},d_{p_2} \neq d_1,d_2$
y_n	$y_n \neq y_1$	$y_{n_1} = y_{n_2} \neq y_1$	$y_{n_1},y_{n_2} \neq y_1,y_2$	$y_{n_1} = y_{n_2} \neq y_1$	$y_{n_1},y_{n_2} \neq y_1,y_2$
d_n	$d_n \neq d_1$	$d_{n_1} = d_{n_2} = d_1$	$d_{n_1} = d_{n_2} = d_1$	$(d_{n_1},d_{n_2}) = (d_1,d_2)$ or (d_2,d_1)	$(d_{n_1},d_{n_2}) = (d_1,d_2)$ or (d_2,d_1)

S1,$\cdots$,S5: five sampling strategies described in Section 3.3; y_j and d_j: the label and domain for the jth sample ($j = 1,2$); y_p and d_p: the label and domain for the positive sample; y_n and d_n: the label and domain for the negative sample.

DOI: 10.52396/JUSTC-2023-0010
JUSTC, **2024**, 54(4): 0404

(S4) If we choose a sample with the same label $y_2 = y_1$ and different domain $d_2 \neq d_1$ as the second original sample, then we choose two samples (label $y_{p_1} = y_{p_2} = y_0$; domains $d_{p_1}, d_{p_2} \neq d_1, d_2$) to mix up a positive sample and we choose two samples (label $y_{n_1} = y_{n_2} \neq y_1$; domains $(d_{n_1}, d_{n_2}) = (d_1, d_2)$ or (d_2, d_1)) to mix up a negative sample.

(S5) If we choose a sample with different label $y_2 \neq y_1$ and different domain $d_2 \neq d_1$ as the original second sample, then we choose two samples (label $(y_{p_1}, y_{p_2}) = (y_1, y_2)$ or (y_2, y_1); domains $d_{p_1}, d_{p_2} \neq d_1, d_2$) to mix up a positive sample and two samples (labels $y_{n_1}, y_{n_2} \neq (y_1, y_2)$; domains$(d_{n_1}, d_{n_2}) = (d_1, d_2)$ or (d_2, d_1))) to mix up a negative sample.

3.4 Feature transfer

In this subsection, we aim to extract a sample feature that is unrelated to the domain and related to the label. Then, we can combine such a feature with a new artificially generated domain feature to obtain a new feature, such that the corresponding original label and the new domain can be easily predicted.

The causal model shown in Fig. 2 serves as the foundation for our feature transfer model. We assume that the input x is dependent on the label variable y, domain variable d, and hidden variable z, and variables y, d, z are mutually independent. Let f_y, f_d, f_z, and f_x be features corresponding to y, d, z, and x, respectively. We wish to use CNN to extract domain-invariant feature $f(x)$, which is independent of d given y and z (i.e., $f(x) \perp d | y, z$). Let $\hat{d}$ be a new domain different from d, and $f_{\hat{d}}$ be a feature corresponding to $\hat{d}$. In our feature transfer model, we need to combine the domain-invariant feature $f(x)$ with the new domain feature $f_{\hat{d}}$. Let $f_{x,\hat{d}}$ be a new feature generated by $f(x)$ and $f_{\hat{d}}$, then the label y and domain $\hat{d}$ can be predicted through the new feature $f_{x,\hat{d}}$. Following this idea, we apply the domain embedding encoder E to transform a random domain input d to the domain feature $e(d)$. In practice, each domain is assigned an abstract number, such as 0, 1, and 2. Each number corresponds to a vector, which represents the domain's feature and is learned during the training process. The random input is a random integer sampled from $0, 1, \cdots, n_d - 1$, where n_d is the total number of domains. The domain embedding encoder is a map mapping the integer to the corresponding vector. A concatenation $(f(x) : e(d))$ is generated by concatenating $f(x)$ and $e(d)$, and the feature transfer T is used to transfer $(f(x) : e(d))$ into a new feature $t(x, d)$. Then, the new feature $t(x, d)$ is sent to a trained classifier to obtain a classification result, with which we can calculate a label classification loss L_{cls2} and a domain classification loss L_{cls3}. L_{cls2} is a cross-entropy loss:

$$L_{\text{cls2}} = -\sum_{i=1}^{n_c} y^i \log(\hat{y}^i) \tag{4}$$

where n_c is the total number of classes, $(y_1, ..., y_{n_c})$ is the one-hot vector of label y with y^i $y^i = 0$ for $i \neq y$ and $y^i = 1$ for $i = y$, and $\hat{y}^i$ is the output probability for i-th class from the classifier. Similarly, L_{cls3} is a cross-entropy loss:

$$L_{\text{cls3}} = -\sum_{i=1}^{n_d} d^i \log(\hat{d}^i) \tag{5}$$

where n_d is the total number of domains, $(d_1, ..., d_{n_d})$ is the one-hot vector of domain d with $d^i = 0$ for $i \neq d$ and $d^i = 1$ for $i = d$, $\hat{d}^i$ is the output probability for ith domain from the

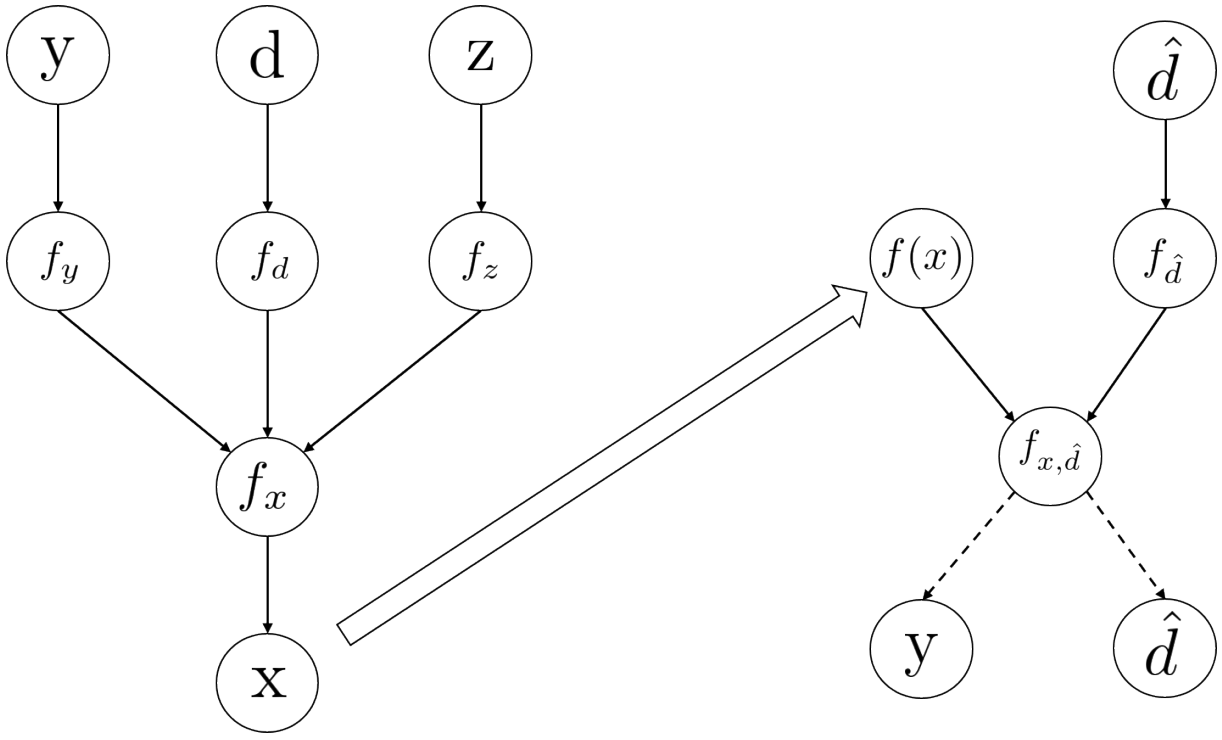


Fig. 2. A causal model underlying our proposed method. y: label variable; f_y: label feature; d: domain variable; f_d: domain feature; z: hidden variable; f_z: hidden feature; x: sample; f_x: sample feature; $f(x)$: domain invariable feature (extracted from CNN); $\hat{d}$: new domain variable; $f_{\hat{d}}$: new domain feature; $f_{x,\hat{d}}$: new feature generated by $f(x)$ and $f_{\hat{d}}$. Solid arrow: causal relationship; dotted arrow: prediction.

DOI: 10.52396/JUSTC-2023-0010
JUSTC, **2024**, 54(4): 0404

classifier.

3.5 Total loss

The total loss is

$$L_{\text{total}} = L_{\text{con}} + \alpha L_{\text{cls1}} + \beta(L_{\text{cls2}} + L_{\text{cls3}}),$$

where L_{con}, L_{cls1}, L_{cls2}, and L_{cls3} are defined in Eqs. (2), (1), (4), and (5), respectively, and α and β are hyper-parameters. Only the contrastive loss, Mixup sampling strategy, and feature transfer model are used during the training and validating stages. In the testing process and applications to real datasets, only the well-trained feature extracting network and the classifier are used to predict labels.

4 Results

4.1 Datasets

We evaluated the performance of the proposed method through the application to the benchmarks PACS[31] and DomainNet[21]. PACS is a widely used domain generalization benchmark consisting of a total of 9991 samples from four domains (i.e., Art Painting, Cartoon, Photo, and Sketch). Each domain includes samples from seven different categories (i.e., dog, elephant, giraffe, guitar, horse, house, and person). DomainNet is a large-scale dataset widely used in multi-source domain adaptation and domain generalization, which consists of 0.6 million images categorized into 345 classes and distributed across six domains (i.e., Clipart, Infograph, Quickdraw, Painting, Real, and Sketch).

4.2 Leave-one-domain-out test strategy

In domain generalization tasks, the leave-one-domain-out strategy is widely used to test the performance of a classification model. The validation schemes are usually implemented as follows:

(Ⅰ) Choose one domain as the target domain and the test dataset consists of all samples with this domain, while the other domains are so-called source domains, and all samples with source domains are divided into a training set and a validation set.

(Ⅱ) Train the model on the training set and choose the model with the best performance on the validation set.

(Ⅲ) Test the performance of the model on the target domain and record the accuracy of the label classification.

(Ⅳ) Treat each domain as a target domain one by one and calculate the average of the accuracies.

4.3 Implementation details

Following the backbone setting of Bai et al.[18], we used ResNet18 and ResNet50 pre-trained on ImageNet as the CNN backbone of our model. The feature used in our model was a concatenation of avgpool, maxpool, and minpool output from the backbone. Domain embedding we adopted was the embedding layer with the input dimension being the source domain number and the output dimension being 256. For ResNet18, the feature transfer module was chosen to be (MLP (1792786), RELU, MLP (7861572)). For ResNet50, the feature transfer module was chosen to be (MLP (6400512), RELU, MLP (5125144)). We randomly chose a strategy from five Mixup sample strategies mentioned in Section 3.3. During training our model, Adam was used as an optimizer with an initial learning rate of 0.0001, the batch size was set to 60, and the total epoch number was chosen to be 20. The learning rate further decayed to 0.00001 in the last 10 epochs.

4.4 Baselines

We compared our method with nine existing domain generalization baselines. (ⅰ) JiGen[32]: a Jigsaw-puzzle-based generalization method, which focuses on the unsupervised task to solve jigsaw puzzles. (ⅱ) MMLD[33]: a method iteratively dividing samples into latent domains via clustering and training the domain-invariant feature extractor shared among the divided latent domains via adversarial learning. (ⅲ) L2A-OT[18]: a method synthesizing extra data from pseudo-novel domains to augment the source domains. (ⅳ) MatchDG[34]: a matching-based algorithm through data augmentation when base objects are observed and objective approximation otherwise. (ⅴ) SagNet[35]: a method forcing the used model to focus more on image contents shared across domains with image styles ignored. (ⅵ) DDAIG[36]: a domain generalization method based on a domain transformation network. (ⅶ) Vanilla: a simple method based on a plain classification model trained on all available source domains using all annotations. (ⅷ) MetaReg[37]: a method using a regularization function in a Learning to Learn (or meta-learning) framework. (ⅸ) Multi-Headed and DMG[38]: a method learning domain-specific masks for generalization on different domains.

4.5 Results

Analysis results for the PACS dataset are presented in Table 2. Our method obtained the highest score on average, and the score of our method was comparable to the highest score in each domain. JiGen[32] performs poorer than ours, which might be due to the fact that JiGen does not handle domain invariance via contrastive loss, though both methods do not

Table 2. Prediction accuracies for the PACS dataset

Method	Photo	Sketch	Cartoon	Art	Average
MMLD	96.09	72.29	77.16	81.28	81.71
JiGen	96.03	71.35	75.25	79.42	80.51
L2A-OT	**96.20**	73.60	78.20	83.30	82.83
DDAIG	95.30	74.70	78.10	**84.20**	83.10
MatchDG	95.93	**77.11**	**80.03**	79.77	83.21
Ours	96.04	74.98	79.97	83.44	**83.61**

DOI: 10.52396/JUSTC-2023-0010
JUSTC, **2024**, 54(4): 0404

alter the constituents of ResNet18 or add any further modules.

Analysis results for the Domainnet dataset are presented in Table 3. With a relatively modest network size (ResNet18), our method achieved the highest score on average, and the score of our method ranks first for half of the test domains. With a pretty wide network size (ResNet50), our method also obtained the best average score. Although our method did not achieve the highest score in the majority of domains, the differences between our scores and the highest score were not large.

In summary, our method outperformed the considered methods in the applications to benchmark datasets with a variety of network sizes and dataset sizes.

4.6 Ablation study

We conducted an ablation study to examine the impact of various factors on the performance of our method with ResNet18 through the applications to using the PACS dataset. We considered three variations of our method: (i) our method without contrastive loss, which removes the contrastive loss but uses the feature transfer model; (ii) our method without Mixup, which removes Mixup together with the Mixup sampling strategy but still uses the contrastive loss; (iii) our method without feature transfer, which removes the feature transfer model together with label classification loss and domain classification loss as parts of the feature transfer model and still uses the contrastive loss and the Mixup sampling strategy; (iv) our method without loss L_{cls2}, which removes the label classification loss (in Eq. (4)) in the feature transfer model; (v) our method without loss L_{cls3}, which removes the domain classification loss (in Eq. (5)) in the feature transfer model.

The prediction accuracies of the ablation study are presented in Table 4. We have the following observations. First, removing contrastive loss from our method resulted in a substantial drop in accuracy. Second, removing Mixup or feature transfer model from our method resulted in a moderate drop in accuracy. Third, removing either the label classification loss or the domain classification loss in the feature transfer model is equivalent to removing the entire feature transfer model. This is because removing the label classification loss will cause the feature transfer model to ignore the features extracted from the CNN and instead focus on the random domain classification, rendering the feature transfer model ineffective. Similarly, removing the domain classification loss will cause the feature transfer model to disregard the domain features, keeping the features extracted from the CNN unchanged and thus rendering the feature transfer model ineffective. In other words, the contrastive loss made a major contribution to the classification performance of our method, and incorporating the Mixup strategy and the feature transfer model further improved the classification performance.

4.7 Further analysis

To demonstrate the functionality of the feature transfer model, we tracked the transferred label accuracy and transferred domain accuracy on the model with the highest label accuracy on validation datasets. The corresponding results are presented in Table 5. The transferred label accuracies were very close to the label accuracies and the transferred domain accuracies were also very high. This indicated that the feature generated from the feature transfer model can be effectively recognized by the trained classifier, demonstrating that the feature transfer model can successfully transfer the feature into a new domain without losing the label information.

5 Conclusions

We present a novel and powerful domain generalization method based on feature transfer and a Mixup sampling strategy. Notably, our method does not make any changes to the popular feature extraction network or add any modules after the network. We introduce a contrastive loss in the method and propose a Mixup sampling strategy to cooperate with it. We also propose a novel feature transfer model to make the extracted feature to be invariant across domains. The applications to experiments on two standard benchmarks demonstrated that our proposed method outperformed several

Table 3. Prediction accuracies for the Domainnet dataset

Method	Clp	Inf	Pnt	Qdr	Rel	Skt	Average
			ResNet18				
Vanilla	56.5	18.4	45.3	12.4	57.9	38.8	38.2
Multi-Headed	55.4	17.5	40.8	11.2	52.9	38.6	36.1
MetaReg	53.6	**21.0**	45.2	10.6	58.4	42.3	38.5
DMG	**60.0**	18.7	44.5	**14.1**	54.7	41.7	39.0
Ours	58.4	20.3	**45.5**	13.1	**58.8**	**42.5**	**39.8**
			ResNet50				
Vanilla	64.0	23.6	51	13.1	64.4	47.7	44.0
Multi-Headed	61.7	21.2	46.8	13.8	58.4	45.4	41.2
MetaReg	59.7	**25.5**	**50.1**	11.5	**64.5**	**50.0**	43.6
DMG	65.2	22.1	50.0	**15.6**	59.6	49	43.6
Ours	**66.2**	23.2	50.0	15.3	63.2	49.5	**44.6**

Clp: Clipart, Inf: Infograph, Pnt: Painting, Qdr: Quickdraw, Rel: Real, and Skt: Sketch.

DOI: 10.52396/JUSTC-2023-0010
JUSTC, **2024**, 54(4): 0404

Table 4. Prediction accuracies for the ablation study using the PACS dataset

Method	Photo	Sketch	Cartoon	Art	Average
Ours w/o contrastive loss	95.19	67.86	74.74	77.85	78.91
Ours w/o Mixup	95.61	73.08	78.19	81.77	82.16
Ours w/o feature transfer	95.75	72.81	77.79	82.09	82.13
Ours w/o loss L_{cls2}	95.59	72.89	76.91	82.17	81.90
Ours w/o loss L_{cls3}	95.51	73.25	77.56	82.52	82.21
Ours	**96.04**	**74.98**	**79.97**	**83.44**	**83.61**

Table 5. Original and transferred accuracies for the PACS dataset

	Photo	Sketch	Cartoon	Art
Label accuracy	96.01	97.15	97.78	97.58
Transferred label accuracy	95.93	97.30	97.53	97.77
Transferred domain accuracy	95.20	95.45	95.34	96.01

considered competitors.

Acknowledgments

This work was supported by National Natural Science Foundation of China (12171451), Anhui Center for Applied Mathematics.

Conflict of Interest

The authors declare that they have no conflict of interest.

Biographies

Yuesong Wang is currently a graduate student under the tutelage of Prof. Hong Zhang at the University of Science and Technology of China. His research interests focus on machine learning.

Hong Zhang is a Full Professor with the University of Science and Technology of China (USTC). He received his Bachelor' s degree in Mathematics and Ph.D. degree in Statistics from USTC in 1997 and 2003, respectively. His major research interests include statistical genetics, causal inference, and machine learning.

References

[1] Shao R, Lan X, Li J, et al. Multi-adversarial discriminative deep domain generalization for face presentation attack detection. In: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). Long Beach, CA, USA: IEEE, **2020**: 10015–10023.

[2] Ouyang C, Chen C, Li S, et al. Causality-inspired single-source domain generalization for medical image segmentation. *IEEE Transactions on Medical Imaging*, **2023**, *42*: 1095–1106.

[3] Guo L L, Pfohl S R, Fries J, et al. Evaluation of domain generalization and adaptation on improving model robustness to temporal dataset shift in clinical medicine. *Scientific Reports*, **2022**, *12*: 2726.

[4] Motiian S, Piccirilli M, Adjeroh D A, et al. Unified deep supervised domain adaptation and generalization. In: 2017 IEEE International Conference on Computer Vision (ICCV). Venice, Italy: IEEE, **2017**: 5716–5726.

[5] Li H, Pan S J, Wang S, et al. Domain generalization with adversarial feature learning. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition. Salt Lake City, UT, USA: IEEE, **2018**: 5400–5409.

[6] Li D, Yang Y, Song Y Z, et al. Learning to generalize: Meta-learning for domain generalization. *Proceedings of the AAAI Conference on Artificial Intelligence*, **2018**, *32* (1).

[7] Mancini M, Bulò S R, Caputo B, et al. Robust place categorization with deep domain generalization. *IEEE Robotics and Automation Letters*, **2018**, *3*: 2093–2100.

[8] Mancini M, Bulò S R, Caputo B, et al. Best sources forward: Domain generalization through source-specific nets. In: 2018 25th IEEE International Conference on Image Processing (ICIP). Athens, Greece: IEEE, **2018**: 1353–1357.

[9] Segu M, Tonioni A, Tombari F. Batch normalization embeddings for deep domain generalization. *Pattern Recognition*, **2023**, *135*: 109115.

[10] Zhang H, Cisse M, Dauphin Y N, et al. Mixup: Beyond empirical risk minimization. arXiv: 1710.09412, **2017.**

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

[12] Nalisnick E, Matsukawa A, Teh Y W, et al. Do deep generative models know what they don't know? arXiv: 1810.09136, **2019**.

[13] Li Y, Tian X, Gong M, et al. Deep domain generalization via conditional invariant adversarial networks. In: Ferrari V, Hebert M, Sminchisescu C, editors. Computer Vision–ECCV 2018. Cham: Springer, **2018**, 11219: 647–663.

[14] Long M, Cao Y, Wang J, et al. Learning transferable features with deep adaptation networks. In: Proceedings of the 32nd International Conference on Machine Learning. New York: ACM, **2015**, 37: 97–105.

[15] Ganin Y, Lempitsky V. Unsupervised domain adaptation by backpropagation. In: Proceedings of the 32nd International Conference on Machine Learning. New York: ACM, **2015**, 37: 1180–1189.

[16] Bousmalis K, Silberman N, Dohan D, et al. Unsupervised pixel-level domain adaptation with generative adversarial networks. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Honolulu, HI, USA: IEEE, **2017**: 95–104.

[17] Xu R, Chen Z, Zuo W, et al. Deep cocktail network: Multi-source unsupervised domain adaptation with category shift. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition. Salt Lake City, UT, USA: IEEE, **2018**: 3964–3973.

[18] Zhou K, Yang Y, Hospedales T, et al. Learning to generate novel domains for domain generalization. In: Vedaldi A, Bischof H, Brox T, editors. Computer Vision–ECCV 2020. Cham: Springer, **2020**,

DOI: 10.52396/JUSTC-2023-0010
JUSTC, **2024**, 54(4): 0404

12361: 561–578.
[19] Bai H, Sun R, Hong L, et al. DecAug: Out-of-distribution generalization via decomposed feature representation and semantic augmentation. *Proceedings of the AAAI Conference on Artificial Intelligence*, **2021**, *35*: 6705–6713.
[20] Chattopadhyay P, Balaji Y, Hoffman J. Learning to balance specificity and invariance for in and out of domain generalization. In: European Conference on Computer Vision. Cham: Springer, **2020**: 301–318.
[21] Chattopadhyay P, Balaji Y, Hoffman J. Learning to balance specificity and invariance for in and out of domain generalization. In: Vedaldi A, Bischof H, Brox T, editors. Computer Vision–ECCV 2020. Cham: Springer, **2020**, 12354: 301–318.
[22] Peng X, Bai Q, Xia X, et al. Moment matching for multi-source domain adaptation. In: 2019 IEEE/CVF International Conference on Computer Vision (ICCV). Seoul, Korea (South): IEEE, **2020**: 1406–1415.
[23] Li D, Zhang J, Yang Y, et al. Episodic training for domain generalization. In: 2019 IEEE/CVF International Conference on Computer Vision (ICCV). Seoul, Korea (South): IEEE, **2020**: 1446–1455.
[24] S Seo, Y Suh, D Kim, G Kim, J Han, and B Han. Learning to optimize domain specific normalization for domain generalization. In: Vedaldi A, Bischof H, Brox T, et al. editors. Computer Vision─ECCV 2020, Cham: Springer,, **2020**: 68–83.
[25] K Zhou, Y Yang, Y Qiao, et al. Domain generalization with mixstyle. arXiv: 2104.02008, **2021**.
[26] Cai Q, Wang Y, Pan Y, et al. Joint contrastive learning with infinite possibilities. In: Proceedings of the 34th International Conference on Neural Information Processing Systems. New York: ACM, **2020**: 12638–12648.
[27] Chen T, Kornblith S, Norouzi M, et al. A simple framework for contrastive learning of visual representations. In: Proceedings of the 37th International Conference on Machine Learning. New York: ACM, **2020**: 1597–1607.
[28] Mitrovic J, McWilliams B, Walker J, et al. Representation learning via invariant causal mechanisms. arXiv: 2010.07922, **2020**.
[29] He K, Fan H, Wu Y, et al. Momentum contrast for unsupervised visual representation learning. In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). Seattle, WA, USA: IEEE, **2020**: 9726–9735.
[30] Wu Z, Xiong Y, Yu S X, et al. Unsupervised feature learning via non-parametric instance discrimination. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition. Salt Lake City, UT, USA: IEEE, **2018**: 3733–3742.
[31] Yao T, Zhang Y, Qiu Z, et al. SeCo: Exploring sequence supervision for unsupervised representation learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, **2021**, *35*: 10656–10664.
[32] Li D, Yang Y, Song Y Z, et al. Deeper, broader and artier domain generalization. In: 2017 IEEE International Conference on Computer Vision (ICCV). Venice, Italy: IEEE, **2017**: 5543–5551.
[33] Carlucci F M, D'Innocente A, Bucci S, et al. Domain generalization by solving jigsaw puzzles. In: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). Long Beach, CA, USA: IEEE, **2020**: 2224–2233.
[34] Matsuura T, Harada T. Domain generalization using a mixture of multiple latent domains. *Proceedings of the AAAI Conference on Artificial Intelligence*, **2020**, *34*: 11749–11756.
[35] D Mahajan, S Tople, and A Sharma. Domain generalization using causal matching. In: Proceedings of the 38th International Conference on Machine Learning. volume 139 of Proceedings of Machine Learning Research. PMLR, **2021**, 139: 7313–7324.
[36] Nam H, Lee H, Park J, et al. Reducing domain gap by reducing style bias. arXiv: 1910.11645, **2019.**
[37] Zhou K, Yang Y, Hospedales T, et al. Deep domain-adversarial image generation for domain generalisation. *Proceedings of the AAAI Conference on Artificial Intelligence*, **2020**, *34*: 13025–13032.
[38] Balaji Y, Sankaranarayanan S, Chellappa R. MetaReg: towards domain generalization using meta-regularization. In: Proceedings of the 32nd International Conference on Neural Information Processing Systems. New York: ACM, **2018**: 1006–1016.
[39] Chattopadhyay P, Balaji Y, Hoffman J. Learning to balance specificity and invariance for in and out of domain generalization. In: Vedaldi A, Bischof H, Brox T, editors. Computer Vision–ECCV 2020. Cham: Springer, **2020**: 301–318.

DOI: 10.52396/JUSTC-2023-0010
JUSTC, **2024**, 54(4): 0404