# Adversarial Learning for Zero-shot Domain Adaptation

Jinghua Wang[0000−0002−2629−1198] and Jianmin Jiang[0000−0002−7576−3999]

Research Institute for Future Media Computing, College of Computer Science & Software Engineering, and Guangdong Laboratory of Artificial Intelligence & Digital Economy (SZ), Shenzhen University, Shenzhen, China.
{wang.jh,jianmin.jiang}@szu.edu.cn†

**Abstract.** Zero-shot domain adaptation (ZSDA) is a category of domain adaptation problems where neither data sample nor label is available for parameter learning in the target domain. With the hypothesis that the shift between a given pair of domains is shared across tasks, we propose a new method for ZSDA by transferring domain shift from an *irrelevant task* ($IrT$) to the *task of interest* ($ToI$). Specifically, we first identify an $IrT$, where dual-domain samples are available, and capture the domain shift with a coupled generative adversarial networks (CoGAN) in this task. Then, we train a CoGAN for the $ToI$ and restrict it to carry the same domain shift as the CoGAN for $IrT$ does. In addition, we introduce a pair of co-training classifiers to regularize the training procedure of CoGAN in the $ToI$. The proposed method not only derives machine learning models for the non-available target-domain data, but also synthesizes the data themselves. We evaluate the proposed method on benchmark datasets and achieve the state-of-the-art performances.

**Keywords:** Transfer Learning; Domain Adaptation; Zero-shot Learning; Coupled Generative Adversarial Networks

## 1 Introduction

When a standard machine learning technique learns a model with the training data and applies the model on the testing data, it implicitly assumes that the testing data share the distribution with the training data [16, 37, 38]. However, this assumption is often violated in applications, as the data in real-world are often from different domains [32]. For example, the images captured by different cameras follow different distributions due to the variations of resolutions, illuminations, and capturing views.

Domain adaptation techniques tackle the problem of domain shift by transferring knowledge from the label-rich source domain to the label-scarce target domain [6, 15, 2]. They have a wide range of applications, such as person re-identification [43], semantic segmentation [24], attribute analysis [44], and medical image analysis [7]. Most domain adaptation techniques assume that the data

† The corresponding author: Jianmin Jiang

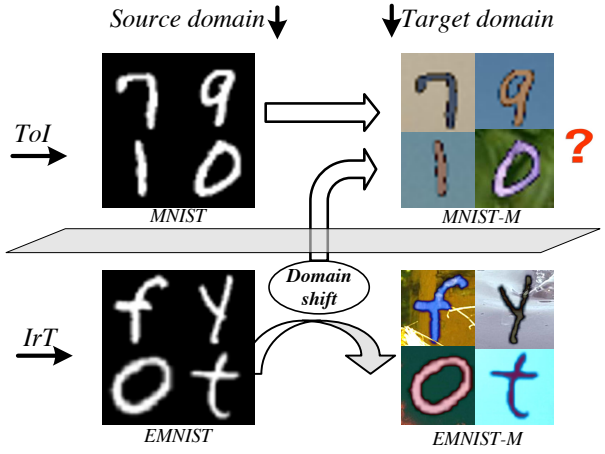



**Fig. 1.** An intuitive example of ZSDA (best viewed in color). The *ToI* is digit image analysis and the *IrT* is letter image analysis. The source domain consists of gray scale images and the target domain consists of color images. In order to learn the model for the unseen *MNIST-M* (*i.e.,* target-domain data in *ToI*), we first learn the domain shift based on the dual-domain samples in the *IrT*, then transfer it to the *ToI*.

in target domain are available at the training time for model learning [42, 23, 6, 29]. However, this is not always the case in the real-world. For example, we may want a artificial intelligence system to provide continuous service with a newly installed camera [19]. This involves a domain adaptation task, where the source domain consists of the images captured by the old camera and the target domain consists of the non-accessible images captured by the new camera. Such a task is referred to as domain generalization [8] or zero-shot domain adaptation (ZSDA) [28, 36].

In this paper, we propose a new method to tackle the challenging ZSDA tasks, where only the source-domain data is available in the *Task of Interest (ToI)*. It is recognized that the existence of domain shift does not allow us to learn a model for the target domain based on the source-domain data alone. To solve this problem, we establish a hypothesis that the domain shift, which intrinsically characterizes the difference between the domains, is shared by different tasks. For successful ZSDA, we firstly learn the domain shift from an *irrelevant task (IrT)* where many data in both domains are available, then transfer this domain shift to the *ToI* and learn the model for the target domain.

We illustrate an example of ZSDA in Fig. 1, which learns a model for the color digit images (*i.e., MNIST-M* [6]), given the grayscale digit images (*i.e., MNIST* [25]), the grayscale letter images (*i.e., EMNIST* [3]), and the color letter images (*i.e., EMNIST-M*). In this example, the *ToI* and *IrT* are digit and letter image analysis, respectively. The source domain consists of grayscale images, and the target domain consists of color images. We consider these two tasks to have the same domain shift, which transforms grayscale images to color images.

With the available dual-domain data in the *IrT*, we can train a coupled generative adversarial networks (CoGAN) [21] (*i.e., CoGAN-IrT*) to model the joint distribution of images in these two domains. This *CoGAN-IrT* not only shows the sharing of two domains in high-level concepts, but also implicitly encodes the difference between them, which is typically referred to as domain shift. We consider one source-domain sample and one target-domain to be *paired samples* if they are realizations of the same thing and correspond to each other. Fig. 1 shows eight grayscale images and their correspondences in the color domain. The RGB image and depth image of the same scene are also paired samples [39]. Based on the observation that it is the domain shift that introduces the difference between *paired samples*, we define the domain shift to be the distribution of representation difference between *paired samples*.

For successful ZSDA in the *ToI*, we train a *CoGAN-ToI* to capture the joint distribution of dual-domain samples and use it to synthesize the unseen samples in the target domain. Besides the available samples in the source domain, we introduce two supervisory signals for *CoGAN-ToI* training. Firstly, we transfer the domain shift from *IrT* to *ToI* and enforce the *CoGAN-ToI* to encode the same domain shift with *CoGAN-IrT*. In other words, we restrict that the representation difference between paired samples to follow the same distribution in two tasks. To improve the quality of the synthesized target-domain samples, we also take a pair of co-training classifiers to guide the training procedure of *CoGAN-ToI*. The predictions of these two classifiers are trained to be (i) consistent when receiving samples from both *IrT* and *ToI*, and (ii) different as much as possible when receiving samples which are not from these two tasks. In the training procedure, we guide the *CoGAN-ToI* to synthesize such target-domain samples that the classifiers produce consistent predictions when taking them as the input. With domain shift preservation and co-training classifiers consistency as the supervisory signals, our *CoGAN-ToI* can synthesize high quality data for the non-accessible target domain and learn well-performed models.

To summary, we propose a new method for ZSDA by learning across both domains and tasks and our contributions can be highlighted in two folds.

- Firstly, we propose a new strategy for domain adaptation through domain shift transferring across tasks. For the first time, we define the domain shift to be the distribution of representation difference between paired samples in two domains. We learn the domain shift from a *CoGAN-IrT* that captures the joint distribution of dual-domain samples in the *IrT* and design a method for shift transferring to the *ToI*, where only source domain is seen. In addition to domain shift preservation, we also take the consistency of two co-training classifiers as another supervisory signal for *CoGAN-IoT* training to better explore the non-accessible target domain.
- Secondly, our method has a broader range of applications than existing methods [28, 36]. While our method is applicable when paired samples in the *IrT* are non-accessible, the work [28] is not. While our method can learn the domain shift from one *IrT* and transfer it to multiple different *ToI*s, the work [36] is only applicable to a given pair of *(IrT, ToI)*.

## 2 Related Work

While standard machine learning methods involves with a single domain [13, 35], domain adaptation uses labeled data samples in one or more source domains to learn a model for the target domain. For transferable knowledge learning, researchers normally minimize the discrepancy between domains by learning domain-invariant features [6, 22, 33, 40]. Ganin and Lempitsky [6] introduced gradient reversal layer to extract features that can confuses the domain classifier. Long *et al.*[22] introduced residual transfer network to bridge the source domain and the target domain by transferable features and adaptive classifiers. Taking maximum mean discrepancy (MMD) as the measurement between domains, Tzeng *et al.*[33] introduced an adaptation layer to learn representations which are not only domain invariant but also semantically meaningful. In order to solve the problem of class weight bias, Yan *et al.*[40] introduced weighted MMD and proposed a classification EM algorithm for unsupervised domain adaptation.

These methods achieve good performances in various computer vision tasks. However, none of them can solve the ZSDA problem as they rely on the target-domain data at the training time. The existing techniques for ZSDA can be summarized into three categories based on their strategies.

The first strategy learns domain-invariant features which not only work in the available source domains but also generalize well to the unseen target domain. Domain-invariant component analysis (DICA) [26] is a kernel-based method that learns a common feature space for different domains while preserving the posterior probability. For cross domain object recognition, multi-task autoencoder (MTAE) [8] extends the standard denoising autoencoder framework by reconstructing the analogs of a given image for all domains. Conditional invariant deep domain generalization (CIDDG) [20] introduces an invariant adversarial network to align the conditional distributions across domains and guarantee the domain-invariance property. With a structured low-rank constraint, deep domain generalization framework (DDG) [5] aligns multiple domain-specific networks to learn sharing knowledge across source domains.

The second strategy assumes that a domain is jointly determined by a sharing latent common factor and a domain specific factor [14, 18, 41]. This strategy identifies the common factor through decomposition and expects it to generalize well in the unseen target domain. Khosla *et al.*[14] model each dataset as a biased observation of the visual world and conduct the decomposition via max-margin learning. Li *et al.*[18] develop a low-rank parameterized CNN model to simultaneously exploit the relationship among domains and learn the domain agnostic classifier. Yang and Hospedales [41] parametrise the domains with continuous values and propose a solution to predict the subspace of the target domain via manifold-valued data regression. Researchers also correlate domains with semantic descriptors [15] or latent domain vectors [17].

The third strategy first learns the correlation between domains from an assistant task, then accomplishes ZSDA based on the available source-domain data and the domain correlation [28, 36]. Normally, this strategy relies on an $IrT$ where data from both source and target domain are sufficiently available.

In comparison with the first two strategies, this strategy can work well with a single source domain. Zero-shot deep domain adaptation (ZDDA) [28] aligns representations from source domain and target domain in the $IrT$ and expect the alignment in the *ToI*. CoCoGAN [36] aligns the representation across tasks in the source domain and takes the alignment as the supervisory signal in the target domain.

## 3 Background

**Generative Adversarial Networks** (GAN) consists of two competing models, *i.e.,* the generator and the discriminator [9]. Taking a random vector $z \sim p_z$ as the input, the generator aims to synthesize images $g(z)$ which are resemble to the real image as much as possible. The discriminator tries to distinguish real images from the synthesized ones. It takes an image $x$ as the input and outputs a scalar $f(x)$, which is expected to be large for real images and small for synthesized images. The following objective function formulates the adversarial training procedure of the generator and the discriminator:

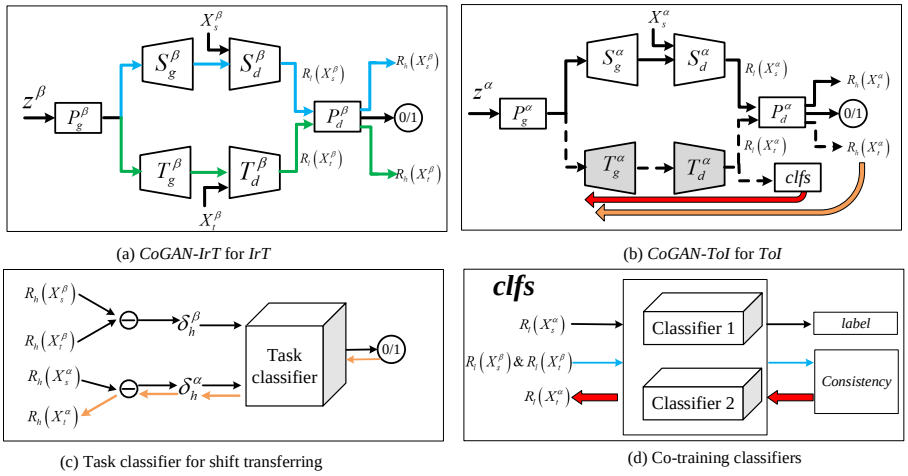
$$\max_{g} \min_{f} V(f,g) \equiv E_{x\sim p_x}[-\log f(x)] + E_{z\sim p_z}[-\log(1-f(g(z)))], \tag{1}$$

where $E$ is the empirical estimate of expected value of the probability. In fact, Eq. (1) measures the Jensen-Shannon divergence between the distribution of real images and that of the synthesized images [9].

**Coupled Generative Adversarial Networks** (CoGAN) consists of a pair of GANs (*i.e.,* $\text{GAN}_1$ and $\text{GAN}_2$) which are closely related with each other. With each GAN corresponds to a domain, CoGAN captures the joint distribution of images from two different domains [21]. Let $x_i \sim p_{x_i} (i = 1, 2)$ be the images from the $i$th domain. In $\text{GAN}_i (i = 1, 2)$, we denote the generator as $g_i$ and the discriminator as $f_i$. Based on a sharing random vector $z$, the generators synthesize image pairs $(g_1(z), g_2(z))$ which not only are indistinguishable from the real ones but also have correspondences. We can formulate the objective function of the CoGAN as follows:

$$\begin{aligned}\max_{g_1,g_2} \min_{f_1,f_2} V(f_1,f_2,g_1,g_2) \equiv & E_{x_1\sim p_{x_1}}[-\log f_1(x_1)] + E_{z\sim p_z}[-\log(1-f_1(g_1(z)))] \\ & + E_{x_2\sim p_{x_2}}[-\log f_2(x_2)] + E_{z\sim p_z}[-\log(1-f_2(g_2(z)))],\end{aligned} \tag{2}$$

subject to two constraints: (i) $\theta_{g_1^j} = \theta_{g_2^j}$, $1 \le j \le n_g$; and (ii) $\theta_{f_1^{n_1-k}} = \theta_{f_2^{n_2-k}}$, $0 \le k \le n_{fs} - 1$. The parameter $n_i (i = 1, 2)$ denotes the number of layers in the discriminator $f_i$. While the first constraint restricts the generators to have $n_g$ sharing bottom layers, the second restricts the discriminators to have $n_f$ sharing top layers. These two constraints force the generators and discriminators to process the high-level concepts in the same way, so that the CoGAN is able to discover the correlation between two domains. CoGAN is effective in dual-domain analysis, as it is capable to learn the joint distribution of data samples (*i.e.,* $p_{x_1,x_2}$) based on the samples drawn individually from the marginal distributions (*i.e.,* $p_{x_1}$ and $p_{x_2}$).




(a) *CoGAN-IrT* for *IrT* (b) *CoGAN-ToI* for *ToI*

(c) Task classifier for shift transferring (d) Co-training classifiers

**Fig. 2.** The network structures of our method. The *CoGAN-IrT* in (a) models the joint distribution of $(x_s^\beta, x_t^\beta)$ in the *IrT*. The *CoGAN-ToI* in (b) models the joint distribution of $(x_s^\alpha, x_t^\alpha)$ in the *ToI*. In the discriminators of these two CoGANs, we use $R_l(\cdot)$ to denote the lower level representation produced by the non-sharing layers, and $R_h(\cdot)$ to denote higher level representations produced by the sharing layers, respectively. The task classifier in (c) discriminates $\delta_h^\beta = R_h(x_t^\beta) \ominus R_h(x_s^\beta)$ from $\delta_h^\alpha = R_h(x_t^\alpha) \ominus R_h(x_s^\alpha)$. We maximize the loss of this task classifier to align the domain shift. The co-training classifiers in (d) produce the labels for $X_s^\alpha$ and consistent predictions for $X_s^\beta$ and $X_t^\beta$. To train the *CoGAN-ToI*, we use domain shift preservation to regularize the higher level features and co-training classifiers to regularize the lower level features. The back-propagation directions of these two signals are marked by orange and red, respectively.

## 4 Approach

### 4.1 Problem Definition

We define a domain $D = \{X, P(X)\}$ to be the data sample space $X$ and its marginal probability distribution $P(X)$ [27]. Given the data samples $X$, a task $T = \{Y, P(Y|X)\}$ consists of a label space $Y$ and the conditional probability distribution $P(Y|X)$. This work considers two tasks to be the same as long as they have sharing label space. In *ToI*, the label space is $Y^\alpha$, the source domain is $D_s^\alpha = \{X_s^\alpha, P(X_s^\alpha)\}$, and the target domain is $D_t^\alpha = \{X_t^\alpha, P(X_t^\alpha)\}$. Then, the *ToI* is denoted by $T^\alpha = \{Y^\alpha, P_s(Y^\alpha|X_s^\alpha)\} \cup \{Y^\alpha, P_t(Y^\alpha|X_t^\alpha)\}$.

Given the labeled data samples in the source domain (*i.e.,* $(x_s^\alpha, y_s^\alpha)$, $x_s^\alpha \in X_s^\alpha$ and $y_s^\alpha \in Y^\alpha$), our ZSDA task aims to derive the conditional probability distribution $P(Y^\alpha|X_t^\alpha)$ in the target domain. In general, the main challenge of this task is induced by non-accessibility of the target domain, as well as the domain shift, *i.e.,* $P(X_s^\alpha) \neq P(X_t^\alpha)$ and $P_s(Y^\alpha|X_s^\alpha) \neq P_t(Y^\alpha|X_t^\alpha)$.

### 4.2 Main Idea

In order to accomplish the ZSDA task, we identify an irrelevant task ($IrT$) that satisfies two constraints: (i) the $IrT$ involves the same pair of domains with the $ToI$; and (ii) the dual-domain samples in the $IrT$ are available. Under the hypothesis that the shift between a given pair of domains maintains across tasks, we propose to learn the domain shift from $IrT$ and transfer it to $ToI$.

Let the label space of $IrT$ be $Y^\beta$. We denote the $IrT$ as $T^\beta = \{Y^\beta, P_s(Y^\beta|X_s^\beta)\} \cup \{Y^\beta, P_t(Y^\beta|X_t^\beta)\}$, where $D_s^\beta = \{X_s^\beta, P(X_s^\beta)\}$ is the source domain and $D_t^\beta = \{X_t^\beta, P(X_t^\beta)\}$ is the target domain, respectively. Note that, the source-domain samples ($X_s^\alpha$ and $X_s^\beta$) are in the same sample space. It is also true in the target domain. In the example of Fig. 1, while the source-domain data $X_s^\alpha$ =*MNIST* and $X_s^\beta$ =*EMNIST* are grayscale images, the target-domain data $X_t^\alpha$ =*MNIST-M* and $X_t^\beta$ =*EMNIST-M* are color images.

In this work, we define two corresponding samples from source domain and target domain as *paired samples*. In most cases, two *paired samples* are different views of the same object. For example, a grayscale image in *MNIST* and its corresponding color image in *MNIST-M* are paired samples in Fig. 1. The depth image and RGB image of the same scene are also *paired samples*. While the similarity between *paired samples* is determined by the object itself, their difference is mainly introduced by the domain shift. Our work only assumes the existences of correspondence between dual-domain samples. Nevertheless, the correspondences between them in the $IrT$ are not required.

For correlation analysis between *paired samples*, we train CoGANs to capture the joint distribution of source-domain and target-domain data. As both $X_s^\beta$ and $X_t^\beta$ are available, we can easily train *CoGAN-IrT* (Fig. 2 (a)) for the $IrT$ using the standard method [21]. The main difficulty lies in the training of *CoGAN-ToI* (Fig. 2 (b)) for the $ToI$, as the target-domain data $X_t^\alpha$ is not available. To tackle this problem, we propose two kinds of supervisory information for *CoGAN-ToI* training, which are *domain shift preservation* and *co-training classifiers consistency*.

For easier transferring across tasks, we define domain shift to be the distribution of element-wise difference between *paired samples* in the representation space. We can learn the domain shift from the *CoGAN-IrT* which carries the correlation between two domains by varying the inputting noise $z^\beta$. After that, we train *CoGAN-ToI* and enforce the representation difference between *paired samples* of $ToI$ to follow the distribution learned in *CoGAN-IrT* by maximizing the loss of a task classifier. Fig. 2 (c) visualizes the task classifier which aims to identify the task label of the representation difference. In this way, the domain shift is transferred from the $IrT$ to $ToI$.

To better explore the unseen target domain of $ToI$, we also build two co-training classifiers (Fig. 2 (d)) and use their consistency to guide the training procedure of *CoGAN-ToI*. By enforcing the weights of the classifiers to be different from each other as much as possible, we aim to analyze data samples from distinct views. The classifiers are trained to: (i) predict the labels of $X_s^\alpha$; (ii) produce consistent predictions when receiving $X_s^\beta$ and $X_t^\beta$; and (iii) produce

different predictions when receiving samples not involved with *ToI* or *IrT*. Thus, we can use the consistency of these two classifiers to evaluate whether a sample is involved with the two tasks. We guide the training procedure of *CoGAN-ToI* to synthesize $X_t^\alpha$ as such that the classifiers also produce consistent predictions when receiving their representations.

### 4.3 Training

In *CoGAN-IrT* (Fig. 2 (a) ), the sharing layers are $P_g^\beta$ and $P_d^\beta$, the non-sharing layers are $S_g^\beta$, $T_g^\beta$, $S_d^\beta$ and $T_d^\beta$. The components of *CoGAN-ToI* are denoted in similar way in Fig. 2 (b). For simplicity, we use $R_l(x)$ to denote the lower level representation of sample $x$ produced by the non-sharing layers and and $R_h(x)$ to denote the higher level representation produced by sharing layers. Note, the representation extraction procedures $R_l(\cdot)$ and $R_h(\cdot)$ vary with task and domain.

**Domain shift**

We train *CoGAN-IrT* based on the dual-domain samples ($X_s^\beta$ and $X_t^\beta$) and let it carry the correlation between two domains. The *CoGAN-IrT* can synthesize a set of *paired samples* for the *IrT*. For two *paired samples* ($x_s^\beta \in X_s^\beta, x_t^\beta \in X_t^\beta$), we characterize their shift by the element-wise difference between representations in a sharing layer, *i.e.,* $\delta_h^\beta = R_h(x_t^\beta) \ominus R_h(x_s^\beta)$. We then define the domain shift to be the distribution of $\delta_h^\beta$, *i.e.,* $p_{\delta_h^\beta}$. Specifically, we can obtain a set of $\{\delta_h^\beta\}$ by feeding *CoGAN-IrT* with different values of inputting noise $z^\beta$.

**Co-training classifiers**

Both of the two co-training classifiers (denoted as $clf_1$ and $clf_2$) take the representations (*i.e.,* $R_l(X_s^\beta)$, $R_l(X_t^\beta)$, and $R_l(X_s^\alpha)$) as the input. With $R_l(x)$ as the input, the classifier $clf_i$ $(i = 1, 2)$ produces a $c$-dimensional vector $v_i(x)$, where $c$ denotes the number of categories in $X_s^\alpha$. We minimize the following loss to train the classifiers:

$$L(clf_1, clf_2) = \lambda_w L_w(w_1, w_2) + \lambda_{acc} L_{cls}(X_s^\alpha) - \lambda_{con} L_{con}(X^\beta) + \lambda_{diff} L_{diff}(\tilde{X}), \quad (3)$$

where $L_w$ measures the similarity between the two classifiers, $L_{cls}$ denotes the loss to classify the labeled source-domain samples of *ToI*, $L_{con}$ assesses the consistency of the output scores when receiving dual-domain samples of *IrT* (*i.e.,* $X_s^\beta$ and $X_t^\beta$) as the input, and $L_{diff}$ assess the consistency when receiving samples $\tilde{X}$ which are not related *ToI* or *IrT*.

As in standard co-training methods [31], we expect the two classifiers to have diverse parameters so that they can analyze the inputs from different views. In this work, we implement these two classifiers with the same neural network structure and assess their similarity by the cosine distance between the parameters:

$$L_w = w_1^T * w_2 \Big/ ||w_1|| * ||w_2|| \,, \quad (4)$$

where $w_i$ is the vectored parameters of $clf_i$.

With the labeled source-domain data in *ToI*, we can easily formulate a multi-class classification problem and use the soft-max loss to define the second term

of Eq. (3) as follows:

$$L_{cls} = -\sum_{x_s^\alpha \in X_s^\alpha} \sum_{i=1}^{2} \sum_{j=1}^{c} v_i^j(x_s^\alpha) l^j(x_s^\alpha), \tag{5}$$

where $v_i^j(x_s^\alpha)$ is the $j$th element of the prediction $v_i(x_s^\alpha)$, the binary value $l^j(x_s^\alpha)$ denotes whether $x_s^\alpha$ belongs to the $j$th class or not. This item regularizes the classifiers to produce semantically meaningful vectors.

Different from $X_s^\alpha$ in *ToI*, the labels for dual-domain data in $IrT$ are not available. It is impossible to predict their true labels. To gain supervisory signals from these label-missing data, we restrict the two classifiers to produce consistent predictions. The consistency for a given sample is measured by the dot product of its two predictions. Thus, we define the third term in Eq. (3) as:

$$L_{con} = \sum_{x^\beta \in X_s^\beta \cup X_t^\beta} v_1(x^\beta) \cdot v_2(x^\beta). \tag{6}$$

The last term $L_{diff}$ regularizes the classifiers to produce different predictions when receiving samples that are not related with the two tasks. It is defined in the same way to $L_{con}$ and the only difference lies in the input $\tilde{X}$. Here, the samples in $\tilde{X}$ have two sources: (i) the samples in public datasets, *e.g.,* imageNet [4]; (ii) the corrupted images by replacing a patch of $x_s^\beta$, $x_t^\beta$, and $x_s^\alpha$ with random noise.

In principal, we can use the consistency of these two classifiers to assess whether a sample is properly involved with $IrT$ or *ToI* in these two domains. Thus, we can guide the training procedure of *CoGAN-ToI* in such a way that the synthesized $X_t^\alpha$ should satisfy $v_1(X_t^\alpha) = v_2(X_t^\alpha)$, and take this as a complementary supervisory signal of domain shift preservation.

**CoGAN-ToI**

At this stage, we train *CoGAN-ToI* to capture the joint distribution of *paired samples* in the *ToI*. By correlating the two domains, a well-trained *CoGAN-ToI* is able to synthesize the non-available target-domain data. We use three constraints to train *CoGAN-ToI*, including (i) one branch captures the distribution of $X_s^\alpha$; (ii) the domain shift is shared by the two tasks, *i.e.,* $p_{\delta_h^\beta} = p_{\delta_h^\alpha}$, where $\delta_h^\alpha = R_h(x_t^\alpha) \ominus R_h(x_s^\alpha)$; and (iii) the co-training classifiers have consistent predictions for the synthesized sample $x_t^\alpha$, *i.e.,* $v_1(x_t^\alpha) = v_2(x_t^\alpha)$.

This work trains the two branches of *CoGAN-ToI* separately, unlike the standard method in [21] that trains them simultaneously. To satisfy the first constraint, we consider the source-domain branch (consisting of $P_g^\alpha$, $S_g^\alpha$, $S_d^\alpha$, and $P_d^\alpha$) as an independent GAN and train it using the available $X_s^\alpha$.

Though involving in different tasks, both $X_t^\beta$ and $X_t^\alpha$ are images from the target domain. Thus, they are composed of the same set of low-level details. In order to mimic the processing method learned in the $IrT$, we initialize the non-sharing components of *CoGAN-ToI* in the target domain as $T_g^\beta \rightarrow T_g^\alpha$ and $T_d^\beta \rightarrow T_d^\alpha$.

After initialization, we use the second and third constraints to train the non-sharing components ($T_g^\alpha$ and $T_d^\alpha$) for the target domain and fine-tune the sharing

components ($P_g^\alpha$ and $P_d^\alpha$). Specifically, we minimize the following loss function:

$$V(P_g^\alpha, T_g^\alpha, T_d^\alpha, P_d^\alpha) \equiv \lambda_{con}^\alpha \sum_{x_t^\alpha = g_t^\alpha(z^\alpha)} v_1(x_t^\alpha) \cdot v_2(x_t^\alpha) - L_{clf}(\delta_h^\alpha, \delta_h^\beta), \tag{7}$$

where $g_t^\alpha = P_g^\alpha + T_g^\alpha$ is the generator. While the first term assesses how the two classifiers agree with each other, the second term assesses how $\delta_h^\alpha$ is distinguishable from $\delta_h^\beta$.

With *CoGAN-ToI*, we train a classifier for the synthesized target-domain data by three steps. Firstly, we train a classifier $\Phi_s(\cdot)$ for the labeled source-domain data. Then, we synthesize a set of paired samples $(x_s^\alpha, x_t^\alpha)$ and use $\Phi_s(\cdot)$ to predict their labels. Finally, we train a classifier $\Phi_t(\cdot)$ for $x_t^\alpha$ with the constraint $\Phi_s(x_s^\alpha) = \Phi_t(x_t^\alpha)$ and evaluate our method using the average accuracy.

## 5 Experiments

### 5.1 Adaptation Across Synthetic Domains

We conduct experiments on four gray image datasets, including MNIST ($D_M$) [25], Fashion-MNIST ($D_F$) [12], NIST ($D_N$) [10], and EMNIST ($D_E$) [3]. Both MNIST [25] and Fashion-MNIST have 70000 images from 10 classes. NIST is imbalance and has more than 40k images from 52 classes. EMNIST has more than 145k images from 26 classes.

These four datasets are in the gray domain (*G–dom*). We create three more domains for evaluation, *i.e.,* the colored domain (*C–dom*), the edge domain (*E–dom*), and the negative domain (*N–dom*). The *C–dom* is created using the method in [6], *i.e.,* combining an image with a random color patch in BSDS500 [1]. We apply canny detector to create *E–dom* and the operation of $I_n = 255 - I$ to create *N–dom*.

**Implementation details**

In order to learn transferable domain shift across tasks, the two CoGANs (*i.e., CoGAN-IrT* and *CoGAN-ToI*) have the same network structure. The two branches inside these CoGANs also share the same structure, and both generators and discriminators have seven layers. We transform the output of the last convolutional layer of discriminator into a column vector before feeding it into a single sigmoid function. The last two layers in generators and the first two layers in discriminators are non-sharing layers for low-level feature processing.

The task classifier has four convolutional layers to identify the task label of its input. We vary the input noise $z^\beta$ of *CoGAN-IrT* to extract $R_h(x)$ and thus obtain a set of $\delta_h^\beta$. The parameters of the task classifier are initialized with zero-centered normal distribution. We adopt the stochastic gradient descent (SGD) method for optimization. The batch size is set to be 128 and the learning rate is set to be 0.0002.

The co-training classifiers are implemented as convolutional neural networks with three fully connected layers, with 200, 50, and $c$, respectively. We use $c$ to

denote the number of categories in $X_s^\alpha$. We set the hyper-parameter as $\lambda_w = 0.01$, $\lambda_{acc} = 1.0$, $\lambda_{con} = 0.5$, and $\lambda_{diff} = 0.5$.

The source-domain branch in *CoGAN-ToI* is firstly trained independently using the available data $X_s^\alpha$. Based on the two supervisory signals, we use back-propagation method to train $T_g^\alpha$ and $T_d^\alpha$. Simultaneously, the $P_g^\alpha$ and $P_d^\alpha$ are fine-tuned. In our experiment, we train the two branches of *CoGAN-ToI* in an iterative manner to obtain the best results.

**Results**

With the above four datasets, we conduct experiments on ten different pairs of (*IrT, ToI*). Note, $D_N$ and $D_E$ are the same task, as both of them consist of letter images. We test four pairs of source domain and target domain, including ($G$–*dom*, $C$–*dom*), ($G$–*dom*, $E$–*dom*), ($C$–*dom*, $G$–*dom*), and ($N$–*dom*, $G$–*dom*).

We take two existing methods as the benchmarks, including ZDDA [28] and CoCoGAN [36]. In addition, we adapt ZDDA by introducing a domain classifier in order to learn from non-corresponding samples and denote it as ZDDA$_{dc}$. We also conduct ablation study by creating the baseline *CTCC*, which only uses Co-Training Classifiers Consistency to train *CoGAN-ToI*.

**Table 1.** The accuracy of different methods with $(source, target) = (G\text{–}dom, C\text{–}dom)$

| *ToI* | $D_M$ | | | $D_F$ | | | $D_N$ | | $D_E$ | |
|---|---|---|---|---|---|---|---|---|---|---|
| *IrT* | $D_F$ | $D_N$ | $D_E$ | $D_M$ | $D_N$ | $D_E$ | $D_M$ | $D_F$ | $D_M$ | $D_F$ |
| ZDDA | 73.2 | 92.0 | 94.8 | 51.6 | 43.9 | 65.3 | 34.3 | 21.9 | 71.2 | 47.0 |
| CoCoGAN | 78.1 | 92.4 | **95.6** | 56.8 | 56.7 | **66.8** | 41.0 | 44.9 | **75.0** | 54.8 |
| ZDDA$_{dc}$ | 69.3 | 79.6 | 80.7 | 50.6 | 42.4 | 62.0 | 29.1 | 20.2 | 49.8 | 46.5 |
| CTCC | 68.5 | 74.9 | 77.6 | 42.0 | 52.9 | 60.9 | 37.0 | 43.6 | 47.3 | 45.2 |
| Ours | **81.2** | **93.3** | 95.0 | **57.4** | **58.7** | 62.0 | **44.6** | **45.5** | 72.4 | **58.9** |

**Table 2.** The accuracy of different methods with $(source, target) = (G\text{–}dom, E\text{–}dom)$

| *ToI* | $D_M$ | | | $D_F$ | | | $D_N$ | | $D_E$ | |
|---|---|---|---|---|---|---|---|---|---|---|
| *IrT* | $D_F$ | $D_N$ | $D_E$ | $D_M$ | $D_N$ | $D_E$ | $D_M$ | $D_F$ | $D_M$ | $D_F$ |
| ZDDA | 72.5 | 91.5 | 93.2 | 54.1 | 54.0 | 65.8 | 42.3 | 28.4 | 73.6 | 50.7 |
| CoCoGAN | 79.6 | **94.9** | 95.4 | 61.5 | 57.5 | 71.0 | 48.0 | 36.3 | 77.9 | 58.6 |
| ZDDA$_{dc}$ | 66.5 | 83.3 | 84.7 | 49.3 | 50.4 | 58.0 | 42.2 | 31.6 | 65.0 | 41.2 |
| CTCC | 65.5 | 73.9 | 80.5 | 44.0 | 40.8 | 37.3 | 40.0 | 31.4 | 57.7 | 48.2 |
| Ours | **81.4** | 93.5 | **96.3** | **63.2** | **58.7** | **72.4** | **49.9** | **38.6** | **78.2** | **61.1** |

As seen in Tab. 1-4, our method achieves the best performance in average. Taking $D_E$ classification as an example, our method outperforms ZDDA [28] by a margin of 8.9%, and outperforms CoCoGAN [36] by a margin of 4.1% when $IrT$ is $D_F$ in Tab. 1. In average, our method performs 7.38% better than ZDDA and 0.69% better than CoCoGAN in Tab. 1.

**Table 3.** The accuracy of different methods with $(source, target) = (C\text{–}dom, G\text{–}dom)$

| *ToI* | $D_M$ | | | $D_F$ | | | $D_N$ | | $D_E$ | |
|---|---|---|---|---|---|---|---|---|---|---|
| *IrT* | $D_F$ | $D_N$ | $D_E$ | $D_M$ | $D_N$ | $D_E$ | $D_M$ | $D_F$ | $D_M$ | $D_F$ |
| ZDDA | 67.4 | 85.7 | 87.6 | 55.1 | 49.2 | 59.5 | 39.6 | 23.7 | 75.5 | 52.0 |
| CoCoGAN | 73.2 | 89.6 | **94.7** | 61.1 | 50.7 | 70.2 | 47.5 | 57.7 | 80.2 | 67.4 |
| $\text{ZDDA}_{dc}$ | 61.5 | 76.7 | 79.9 | 51.2 | 46.1 | 53.4 | 31.3 | 20.4 | 61.2 | 42.2 |
| CTCC | 62.1 | 76.9 | 68.6 | 47.2 | 45.6 | 57.6 | 27.5 | 33.6 | 58.0 | 49.9 |
| Ours | **73.7** | **91.0** | 93.4 | **62.4** | **53.5** | **71.5** | **50.6** | **58.1** | **83.5** | **70.9** |

**Table 4.** The accuracy of different methods with $(source, target) = (N\text{–}dom, G\text{–}dom)$

| *ToI* | $D_M$ | | | $D_F$ | | | $D_N$ | | $D_E$ | |
|---|---|---|---|---|---|---|---|---|---|---|
| *IrT* | $D_F$ | $D_N$ | $D_E$ | $D_M$ | $D_N$ | $D_E$ | $D_M$ | $D_F$ | $D_M$ | $D_F$ |
| ZDDA | 78.5 | 90.7 | 87.6 | 56.6 | 57.1 | 67.1 | 34.1 | 39.5 | 67.7 | 45.5 |
| CoCoGAN | 80.1 | 92.8 | 93.6 | 63.4 | 61.0 | 72.8 | 47.0 | 43.9 | 78.8 | 58.4 |
| $\text{ZDDA}_{dc}$ | 68.4 | 79.8 | 82.5 | 48.1 | 46.2 | 64.6 | 28.6 | 34.4 | 61.8 | 36.2 |
| CTCC | 68.4 | 80.0 | 80.2 | 50.1 | 55.1 | 61.3 | 37.6 | 33.9 | 56.1 | 33.9 |
| Ours | **82.6** | **94.6** | **95.8** | **67.0** | **68.2** | **77.9** | **51.1** | **44.2** | **79.7** | **62.2** |

In each of the Tab. 1-4, the proposed method improves CTCC more than 10% in average. This means that the domain shift transferring are useful in the training procedure of *CoGAN-ToI*. Among the three tasks, the digit image classification is the easiest one. Out of all settings, the most successful one transfers knowledge from the *G–dom* to the *E–dom* with $D_E$ as the $IrT$ and $D_M$ as the *ToI*. In this case, our method achieves the accuracy of 96.3%. For $D_M$ classification in *G–dom*, our method achieve the accuracy of 95.8% with $D_E$ as $IrT$ and *N–dom* as the source domain, outperforming other techniques (including 89.5% in [11], and 94.2% in [30]) which rely on the availability of the target-domain data in the training stage. With *CoGAN-ToI*, we not only derive models for the unseen target domain, but also synthesize data themselves. Fig. 3 visualizes the generated images in *C–dom* and *E–dom* with *G–dom* as the source domain.

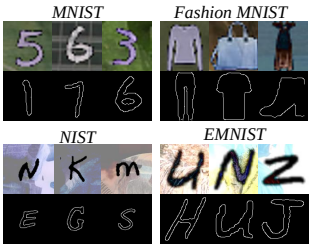



**Fig. 3.** The generated images in the *C–dom* and *E–dom*.

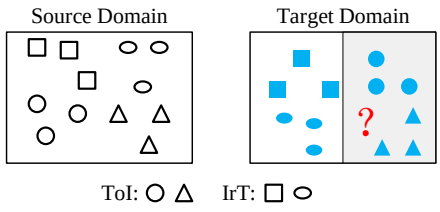



**Fig. 4.** An example. *ToI* represents a subset of the categories (*square* and *triangle*) and $IrT$ represents the rest (*circle* and *ellipse*).

### 5.2 Adaptation in Public Dataset

We also evaluate our method on Office-Home [34], which has four different domains, *i.e.,* Art (**Ar**), Clipart (**Cl**), Product (**Pr**), and Real-world (**Rw**). It has more than 15k images from 65 categories.

As it is difficult to identify an analogous set for this dataset, we evaluate our method on adaptation across subsets. Give a pair of domains, we take a subset of the categories as the *ToI* and the rest as the *IrT*. An example is shown Fig. 4, where the *ToI* represents the classification of two categories (*square* and *triangle*), and *IrT* represents the classification of other two categories (*circle* and *ellipse*). Here, we set the parameters as $\lambda_w = 0.01$, $\lambda_{acc} = 1$, and $\lambda_{con} = \lambda_{diff} = 0.1$.

**Table 5.** The accuracy of different methods on Office-Home

| Source | *Ar* | | | *Cl* | | | *Pr* | | | *Rw* | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Target | *Cl* | *Pr* | *Rw* | *Ar* | *Pr* | *Rw* | *Ar* | *Cl* | *Rw* | *Ar* | *Cl* | *Pr* |
| ZDDA$_{dc}$ | 53.2 | 61.4 | 68.8 | 67.4 | 57.0 | 68.4 | 60.9 | 40.6 | 62.4 | 68.1 | 43.4 | 50.3 |
| CoCoGAN | 62.2 | 69.5 | 74.5 | 66.7 | 74.0 | 66.4 | 57.6 | 53.4 | 71.7 | 69.2 | 51.3 | 65.8 |
| CTCC | 55.7 | 61.5 | 66.5 | 66.8 | 64.6 | 65.2 | 56.3 | 46.6 | 61.6 | 64.3 | 43.7 | 57.7 |
| Ours | **62.7** | **71.9** | **76.3** | **72.6** | **75.1** | **73.9** | **70.3** | **60.8** | **74.8** | **72.2** | **61.4** | **72.2** |

Let $N_\alpha$ denote the number of categories of *ToI*. We fix the value of $N_\alpha$ to be 10 and conduct experiments on all of the 12 possible different pairs of source domain and target domain. As seen in Tab. 5, our method achieves the best performances in all cases. This indicates that our method is applicable to a broad range of applications. Our method can beats both ZDDA and CoCoGAN by a margin larger than 10%, when source domain is *Rw* and target domain is *Cl*.

**Table 6.** The variation of accuracy against parameter $\lambda^\alpha_{con}$

| $\lambda^\alpha_{con}$ | ar→ cl | ar→ pr | ar→ rw | cl→ ar | cl→ pr | cl→ rw |
|---|---|---|---|---|---|---|
| 0.001 | 59.3 | 68.5 | 73.3 | 65.7 | 68.3 | 69.3 |
| 0.005 | 61.6 | 70.3 | 75.7 | 70.6 | 74.6 | 71.1 |
| 0.01 | 62.7 | 71.9 | 76.3 | 72.6 | 75.1 | 73.9 |
| 0.02 | 62.1 | 71.0 | 74.7 | 72.1 | 76.1 | 72.8 |
| 0.1 | 53.0 | 64.8 | 66.1 | 60.4 | 60.4 | 63.5 |

We use the parameter $\lambda^\alpha_{con} = 0.01$ to balance the two terms in Eq. (7). Generally, the CTCC mainly regularizes the training of $T^\alpha_s$, which processes the low-level details. The detail-richer $X^\beta_t$ means more knowledge are available for training, and the more transferable across tasks the $T^\alpha_s$ is. Thus, we set smaller value for $\lambda^\alpha_{con}$ when richer details are included in $X^\beta_t$. Tab. 6 lists the accuracy

of our method on Office-Home with different parameter values of $\lambda_{con}^{\alpha}$. As seen, our method performs well when $\lambda_{con}^{\alpha} \in [0.005, 0.01]$.

Let $N_s$ be the number of samples in the $X = \{X_s^{\alpha}, X_s^{\beta}, X_t^{\beta}\}$. We use $2N_s$ supplementary samples to train the $L_{diff}$ where (i) half are randomly cropped from the ImageNet and (ii) half are obtained by replacing patches of training samples with random noises. Tab. 7 lists the performance of our method with different number of supplementary samples. As seen, $2N_s$ supplementary samples are enough for model training.

**Table 7.** The variation of accuracy against number of supplementary samples

| Num | ar→ cl | ar→ pr | ar→ rw | cl→ ar | cl→ pr | cl→ rw |
|---|---|---|---|---|---|---|
| 0.8N | 60.3 | 67.5 | 73.4 | 68.8 | 67.0 | 70.7 |
| N | 61.3 | 70.7 | 73.6 | 70.3 | 73.2 | 71.0 |
| 1.6N | 62.5 | 71.5 | 76.0 | 71.5 | 74.3 | 73.5 |
| 2N | 62.7 | 71.9 | 76.3 | 72.6 | 75.1 | 73.9 |
| 4N | 62.7 | 71.9 | 76.3 | 72.7 | 75.1 | 73.9 |

## 6 Conclusion and Future Work

This paper proposes a new method for ZSDA based on the hypothesis that different tasks may share the domain shift for the given two domains. We learn the domain shift from one task and transfer it to the other by bridging two CoGANs with a task classifier. Our method takes the domain shift as the distribution of the representation difference between paired samples and transfers it across CoGANs. Our method is capable of not only learning the machine learning models for the unseen target domain, but also generate target-domain data samples. Experimental results on six datasets show the effectiveness of our method in transferring knowledge among images in different domains and tasks.

The proposed method learns the shift between domains and transfers it across tasks. This strategy makes our method to be applicable only when "large" shift exists across domains, such as (rgb, gray), (clipart, art) etc. Thus, our method cannot perform well on the datasets where the domain shift is "small", such as VLSC and Office-31. In the future, we will train a classifier to determine whether correspondence exists between a source-domain sample and a synthesized target-domain sample. Such a classifier can guide the training procedure of CoGAN, even when only samples from a single domain is available.

## Acknowledgment

The authors wish to acknowledge the financial support from: (i) Natural Science Foundation China (NSFC) under the Grant no. 61620106008 ; (ii) Natural Science Foundation China (NSFC) under the Grant no. 61802266.

## References


1. Arbelaez, P., Maire, M., Fowlkes, C., Malik, J.: Contour detection and hierarchical image segmentation. IEEE Trans. Pattern Anal. Mach. Intell. **33**(5), 898–916 (2011)
2. Chen, Y., Lin, Y., Yang, M., Huang, J.: Crdoco: Pixel-level domain transfer with cross-domain consistency. In: CVPR. pp. 1791–1800 (2019)
3. Cohen, G., Afshar, S., Tapson, J., van Schaik, A.: EMNIST: an extension of MNIST to handwritten letters. arXiv (2017)
4. Deng, J., Dong, W., Socher, R., Li, L., Li, K., Li, F.: Imagenet: A large-scale hierarchical image database. In: CVPR. pp. 248–255 (2009)
5. Ding, Z., Fu, Y.: Deep domain generalization with structured low-rank constraint. IEEE Transactions on Image Processing **27**(1), 304–313 (2018)
6. Ganin, Y., Lempitsky, V.: Unsupervised domain adaptation by backpropagation. In: ICML. vol. 37, pp. 1180–1189 (2015)
7. Ghassami, A., Kiyavash, N., Huang, B., Zhang, K.: Multi-domain causal structure learning in linear systems. In: NeurIPS. pp. 6269–6279 (2018)
8. Ghifary, M., Kleijn, W.B., Zhang, M., Balduzzi, D.: Domain generalization for object recognition with multi-task autoencoders. In: ICCV (2015)
9. Goodfellow, I.J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. In: NIPS. pp. 2672–2680 (2014)
10. Grother, P., Hanaoka, K.: Nist special database 19 handprinted forms and characters database. In: National Institute of Standards and Technology (2016)
11. Haeusser, P., Frerix, T., Mordvintsev, A., Cremers, D.: Associative domain adaptation. In: ICCV. pp. 2784–2792 (2017)
12. Han, X., Kashif, R., Roland, V.: Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. CoRR **abs/1708.07747** (2017)
13. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR. pp. 770–778 (2016)
14. Khosla, A., Zhou, T., Malisiewicz, T., Efros, A.A., Torralba, A.: Undoing the damage of dataset bias. In: ECCV (2012)
15. Kodirov, E., Xiang, T., Fu, Z., Gong, S.: Unsupervised domain adaptation for zero-shot learning. In: ICCV (2015)
16. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: NIPS. pp. 1106–1114 (2012)
17. Kumagai, A., Iwata, T.: Zero-shot domain adaptation without domain semantic descriptors. CoRR **abs/1807.02927** (2018)
18. Li, D., Yang, Y., Song, Y.Z., Hospedales, T.M.: Deeper, broader and artier domain generalization. In: ICCV (2017)
19. Li, D., Yang, Y., Song, Y.Z., Hospedales, T.M.: Learning to generalize: Meta-learning for domain generalization. In: AAAI (2018)
20. Li, Y., Tian, X., Gong, M., Liu, Y., Liu, T., Zhang, K., Tao, D.: Deep domain generalization via conditional invariant adversarial networks. In: ECCV (2018)
21. Liu, M.Y., Tuzel, O.: Coupled generative adversarial networks. In: NIPS (2016)
22. Long, M., Zhu, H., Wang, J., Jordan, M.I.: Unsupervised domain adaptation with residual transfer networks. In: NIPS. pp. 136–144 (2016)
23. Lopez-Paz, D., Hernández-Lobato, J., Schölkopf, B.: Semi-supervised domain adaptation with non-parametric copulas. In: NIPS (2012)

24. Luo, Y., Zheng, L., Guan, T., Yu, J., Yang, Y.: Taking a closer look at domain shift: Category-level adversaries for semantics consistent domain adaptation. In: CVPR (2019)
25. Lécun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proceedings of the IEEE (1998)
26. Muandet, K., Balduzzi, D., Schölkopf, B.: Domain generalization via invariant feature representation. In: ICML (2013)
27. Pan, S.J., Yang, Q.: A survey on transfer learning. IEEE TKDE **22**(10), 1345–1359 (2010)
28. Peng, K.C., Wu, Z., Ernst, J.: Zero-shot deep domain adaptation. In: ECCV (2018)
29. Pinheiro, P.O.: Unsupervised domain adaptation with similarity learning. In: CVPR (2018)
30. Saito, K., Ushiku, Y., Harada, T.: Asymmetric tri-training for unsupervised domain adaptation. In: ICML. pp. 2988–2997 (2017)
31. Saito, K., Watanabe, K., Ushiku, Y., Harada, T.: Maximum classifier discrepancy for unsupervised domain adaptation. In: CVPR. pp. 3723–3732 (2018)
32. Torralba, A., Efros, A.A.: Unbiased look at dataset bias. In: CVPR (2011)
33. Tzeng, E., Hoffman, J., N., Z., S., K., D., T.: Deep domain confusion: Maximizing for domain invariance. Computer Science (2014)
34. Venkateswara, H., Eusebio, J., Chakraborty, S., Panchanathan, S.: Deep hashing network for unsupervised domain adaptation. In: CVPR. pp. 5385–5394 (2017)
35. Wang, J., Jiang, J.: An unsupervised deep learning framework via integrated optimization of representation learning and gmm-based modeling. In: ACCV. vol. 11361, pp. 249–265 (2018)
36. Wang, J., Jiang, J.: Conditional coupled generative adversarial networks for zero-shot domain adaptation. In: ICCV (2019)
37. Wang, J., Jiang, J.: Sa-net: A deep spectral analysis network for image clustering. Neurocomputing **383**, 10–23 (2020)
38. Wang, J., Wang, G.: Hierarchical spatial sum-product networks for action recognition in still images. IEEE Trans. Circuits Syst. Video Techn. **28**(1), 90–100 (2018)
39. Wang, J., Wang, Z., Tao, D., See, S., Wang, G.: Learning common and specific features for RGB-D semantic segmentation with deconvolutional networks. In: ECCV. pp. 664–679 (2016)
40. Yan, H., Ding, Y., Li, P., Wang, Q., Xu, Y., Zuo, W.: Mind the class weight bias: Weighted maximum mean discrepancy for unsupervised domain adaptation. In: IEEE. pp. 945–954 (2017)
41. Yang, Y., Hospedales, T.: Zero-shot domain adaptation via kernel regression on the grassmannian (2015). https://doi.org/10.5244/C.29.DIFFCV.1
42. Yao, T., Pan, Y., Ngo, C.W., Li, H., Tao, M.: Semi-supervised domain adaptation with subspace learning for visual recognition. In: CVPR (2015)
43. Zhong, Z., Zheng, L., Luo, Z., Li, S., Yang, Y.: Invariance matters: Exemplar memory for domain adaptive person re-identification. In: CVPR (2019)
44. Zhu, P., Wang, H., Saligrama, V.: Learning classifiers for target domain with limited or no labels. In: ICML. pp. 7643–7653 (2019)