



Unsupervised Domain Adaptation with Joint Domain-Adversarial Reconstruction Networks

Qian Chen, Yuntao Du, Zhiwen Tan, Yi Zhang, and Chongjun Wang^(✉)

State Key Laboratory for Novel Software Technology, Nanjing University,
Nanjing, China

{chenqian, duyuntao, njuzhangyi}@smail.nju.edu.cn,
yaoyueduzhen@outlook.com, chjwang@nju.edu.cn

Abstract. Unsupervised Domain Adaptation (UDA) attempts to transfer knowledge from a labeled source domain to an unlabeled target domain. Recently, domain-adversarial learning has become an increasingly popular method to tackle this task, which bridges source domain and target domain by adversarially learning domain-invariant representations that cannot be discriminated by a domain discriminator. In spite of the great success achieved by domain-adversarial learning, most of existing methods still suffer two major limitations: (1) due to focusing only on learning domain-invariant representations, they ignore the individual characteristics of each domain and fail to extract domain-specific information that is beneficial for final classification; (2) by focusing only on performing domain-level distribution alignment to learn domain-invariant representations, they fail to achieve the invariance of representations at a class level, which may lead to incorrect distribution alignment. To address the above issues, we propose in this paper a novel model called *Joint Domain-Adversarial Reconstruction Network* (JDARN), which integrates domain-adversarial learning with data reconstruction to learn both domain-invariant and domain-specific representations. Meanwhile, we propose to employ two novel discriminators called *joint domain-class discriminators* to achieve the joint alignment and adopt a novel joint adversarial loss to train them. With both domain and class information of two domains, the two discriminators can be used to promote domain-invariant representation learning towards the class level, not only the domain level. Extensive experimental results reveal that the proposed JDARN exceeds the state-of-the-art performance on two standard UDA datasets.

Keywords: Unsupervised domain adaptation · Domain-adversarial learning · Data reconstruction · Distribution alignment

1 Introduction

Deep learning methods have achieved great success in many fields, such as computer vision and nature language process. The access to massive amounts of

labeled training data is one of the reasons for such success. Generally, deep learning methods usually train deep neural networks with a large scale labeled training dataset, and then test on a testing dataset, which has similar distribution as the training one. Nevertheless, training datasets are usually either difficult to collect, or prohibitive to annotate. Meanwhile, due to dataset bias [35] or domain shift [29], traditional deep models do not generalize well on new datasets or tasks.

Unsupervised Domain Adaptation tackles the aforementioned problems by transferring knowledge from a label-rich source domain to a label-scarce target domain whose distribution is different from the source one [25,26]. The deep UDA methods [6,19,21,32,37] have achieved remarkable performance, which leverage deep networks to learn transferable representations by embedding adaptation modules in deep architectures.

Recently, adversarial learning has been successfully embedded into deep networks to learn domain-invariant representations to reduce distribution discrepancy between source domain and target domain. Inspired by generative adversarial networks (GAN) [9], domain-adversarial learning [6] pits two networks against each other—feature extractor and domain discriminator. It plays a minimax game to learn the domain discriminator that aims to distinguish feature representations of source samples from those of target samples, and the feature extractor that aims to learn domain-invariant representations to fool the domain discriminator. Theoretically, domain alignment is achieved when the minimax optimization reach an equilibrium.

In spite of the great success achieved by domain-adversarial learning, most prior efforts still suffer two major limitations. Firstly, most of them only concentrate on learning domain-invariant representations, but ignore the individual characteristics of each domain, and are limited in extracting domain-specific information. Since domain-specific representations not only preserve the discriminability, but also contains more meaningful information of each domain, they are beneficial for the final classification. Secondly, most of them only focus on aligning the marginal distributions of two domains, which is referred as domain-level distribution alignment, but ignore the alignment of class conditional distributions across domains, which is referred as class-level distribution alignment. A perfect domain-level distribution alignment does not imply a fine-grained class-to-class overlap. With an intuitive example shown in Fig. 1, with domain-level alignment only, the learned domain-invariant representations may not only bring source domain and target domain closer, but also mix samples with different class labels together, which may cause false classification. The lack of class-level distribution alignment is a major cause of performance reduction [24]. Thus, it is necessary to pursue the class-level and domain-level distribution alignments simultaneously under the absence of target true labels.

For the first issue, we note that some UDA methods based on encoder-decoder reconstruction [3,8] use data reconstruction of source or target domain samples as an auxiliary task, which typically learn domain-invariant representations by a shared encoder and maintain domain-specific representations

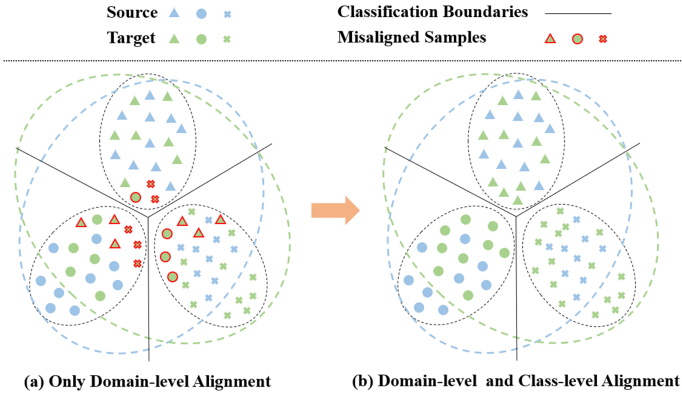


Fig. 1. Left: Only with domain-level alignment, there may exist some misaligned samples in the target domain, which will cause false classification. **Right:** With both domain-level and class-level alignment, the classification performance will be better.

by a reconstruction loss [38]. Inspired by these methods, we integrate the domain-adversarial learning with the data reconstruction to learn domain-invariant and domain-specific representations simultaneously. Besides, most non-reconstruction methods map target domain samples to source domain in the deep feature space, or map source and target domain samples to a common deep feature space, which can not promise that the structure of feature space is not distorted after mapped. On the contrary, reconstruction loss enforces the data decoded from latent representations as close to original data as possible, which is helpful for maintaining the structure of feature space and making it undistorted. In this paper, we choose variational autoencoder (VAE) [14] as the basic encoder-decoder reconstruction model. VAE is a directed graphical model with certain types of latent variables, such as Gaussian latent variables. To some degree, using VAE as the basic reconstruction model is also helpful for alleviating the distribution discrepancy between domains, since the distributions of latent representations from two domains are forced to be close to a same certain distribution by VAE.

For the second issue, to promote domain-invariant representations learning towards the class level, we propose a novel domain-adversarial learning paradigm in this paper. In previous adversarial methods, the output of the domain discriminator is a layer with 2 nodes that can only indicate the domain label of one feature representation. Different from them, in our method, the output of the discriminator is a layer with $K + 1$ (K is the number of classes) nodes, where the first K nodes represent the class of one feature representation and the last node indicates the domain of the representation. We call this new discriminator a *joint domain-class discriminator*, since it learns a joint distribution over both domain and class variables. In our model, we employ two joint domain-class discriminators in our model, and adopt a novel joint adversarial loss to train them, instead of a simple binary adversarial loss. Class labels are needed

when calculating the loss, but they are not available in the target domain. Some existing methods [20, 28, 39] use pseudo-labels to make up for the lack of target domain class labels. Following this idea, we first train a source classifier using labeled source data to initialize a target classifier, and then update the target classifier by a semi-supervised learning (SSL) style [23] to provide target domain pseudo-labels.

Briefly, we summarize our contributions below.

- We propose in this paper a novel model termed Joint Domain-Adversarial Reconstruction Network for unsupervised domain adaptation. Our proposed JDARN integrates domain-adversarial learning with data reconstruction to learn both domain-invariant and domain-specific representations.
- In order to achieve the invariance of representations at the domain level and class level simultaneously, we introduce two novel discriminators called joint domain-class discriminators to our proposed model and adopt a joint adversarial loss to train them.
- We conduct careful experiments to investigate the efficacy of our proposed model. Based on commonly used basic networks (ResNet-50 [11]), our proposed model outperforms the state-of-the-art methods on two benchmark domain adaptation datasets.

2 Related Works

In this section, we briefly review recent domain adaptation methods, in particular domain-adversarial learning methods and those based on data reconstruction.

Inspired by GANs, adversarial learning has been widely adopted in domain adaptation [6, 7, 13, 18, 36, 37]. The domain-adversarial neural network (DANN) [6] uses adversarial training to find domain-invariant representations. DANN minimizes the domain confusion loss for all samples and label prediction loss only for source samples while maximizing domain confusion loss via a gradient reversal layer (GRL). The adversarial discriminative domain adaptation (ADDA) [37] summarizes that each adversarial method makes three design choices: whether to use a generative or discriminative base model, whether to tie or untie the weights, and which adversarial learning objective to use. The ADDA uses a discriminative base model without the generator and unshared weights that can better model the difference in low level features than shared ones. Besides, it use a standard GAN loss function, which has the same fixed-point properties as the minimax loss but provides stronger gradients to the target mapping.

Some recent adversarial UDA methods [20, 28, 31, 33, 39, 40] have paid attention to pursuing the class-level alignment. The moving semantic transfer network (MSTN) [39] learns semantic representations for unlabeled target samples by aligning labeled source centroid and pseudo-labeled target centroid. The multi-adversarial domain adaptation (MADA) [28] and the conditional domain adversarial network (CDAN) [20] exploit multiplicative interactions between feature representations and category predictions as highorder features to help adversarial training.

In another line of UDA methods, data reconstruction is used as an auxiliary task that simultaneously learns shared representations between domains and keeps the individual characteristics of each domain [3, 8, 12]. The deep reconstruction classification network (DRCN) [8] learns a shared encoding representation that provides useful information for cross-domain object recognition. DRCN minimizes the classification loss with labeled source data and the unsupervised reconstruction loss with unlabeled target data. The domain separation network (DSN) [3] explicitly and jointly models both private and shared components of the domain representations to reconstruct the images from both domains.

3 Method

3.1 Problem Setting

For clear description, we give some definitions. In unsupervised domain adaptation, we have access to a source domain $\mathcal{D}_s = \{(\mathbf{x}_s^{(i)}, y_s^{(i)})\}_{i=1}^{N_s}$ of N_s labeled samples from $\mathcal{X}_s \times \mathcal{Y}_s$ and a target domain $\mathcal{D}_t = \{\mathbf{x}_t^{(i)}\}_{i=1}^{N_t}$ of N_t unlabeled samples from \mathcal{X}_t , where \mathcal{X} represents the feature space and \mathcal{Y} represents the label space. The source and target domain samples are sampled from joint distributions $P_s(\mathbf{x}_s, y_s)$ and $P_t(\mathbf{x}_t, y_t)$ respectively, and the i.i.d. assumption is violated as $P_s \neq P_t$. The goal of UDA is to train a deep network $f : \mathbf{x} \rightarrow y$ which formally reduces the shifts in the data distributions across domains, such that the target error $\epsilon_t(f) = \mathbb{E}_{(\mathbf{x}_t, y_t) \sim P_t} [f(\mathbf{x}_t) \neq y_t]$ can be bounded by the sum of the source error $\epsilon_s(f) = \mathbb{E}_{(\mathbf{x}_s, y_s) \sim P_s} [f(\mathbf{x}_s) \neq y_s]$ and the distribution discrepancy $d(P_s, P_t)$.

3.2 Network Structure

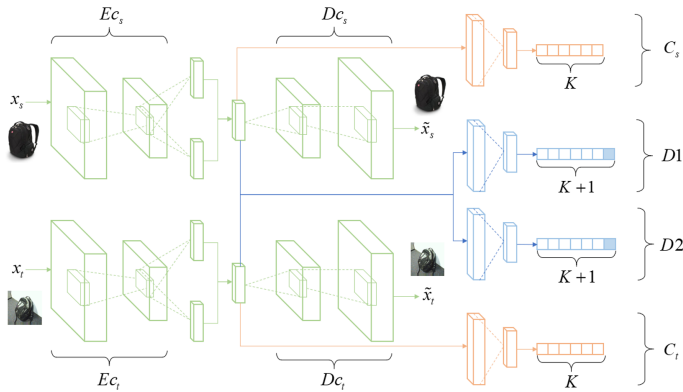


Fig. 2. Overall structure of our proposed JDARN

Our proposed JDARN is equipped with two VAEs, two joint domain-class discriminators and two task-specific classifiers. An overview of the proposed model

is shown in Fig. 2. The ADDA [37] suggests that allowing independent source and target mappings is a more flexible learning paradigm as it allows more domain specific feature extraction to be learned. Therefore, following this idea, we employ two separate VAEs, one for source domain, and another for target domain. The encoders of VAEs, denoted as Ec_s for source and Ec_t for target, encode source domain or target domain samples \mathbf{x} into latent representations $\mathbf{z} = Ec(\mathbf{x})$, which aims to learn domain-invariant representations to fool the two discriminators. The decoders of VAEs, denoted as Dc_s for source and Dc_t for target, decode the latent representations \mathbf{z} back to a reconstructed version, which aims to extract more domain-specific information and make the structure of source and target domain feature space undistorted. The two joint domain-class discriminators, denoted as $D1$ and $D2$, are not only trained to distinguish whether latent representations are from source domain or target domain, but also trained to discriminate the class of the latent representations. The two task-specific classifiers, denoted as C_s for source and C_t for target, are trained to classify the samples from two domains.

3.3 Loss Functions

In this paper, our final goal is to learn a target encoder Ec_t , and a classifier C_t that can classify target domain samples into one of K categories correctly when testing. Due to lack of target domain class labels, it is impossible to directly learn a target encoder and a target classifier. Thereby, we instead train a source encoder, Ec_s , along with a source classifier, C_s , and then adapt that model for use in the target domain.

Classification Loss. First of all, Ec_s and C_s are pre-trained using a standard supervised loss as follows:

$$\mathcal{L}_{cls} = \mathbb{E}_{(\mathbf{x}_s, y_s) \sim \mathcal{D}_s} \ell_{ce}(C_s(Ec_s(\mathbf{x}_s)), y_s), \quad (1)$$

where $\ell_{ce}(f(\mathbf{x}), y) := -\langle y, \log f(\mathbf{x}) \rangle$ is the cross-entropy loss calculated for one-hot, ground-truth label $y \in \{0, 1\}^K$ and label prediction $f(\mathbf{x}) \in \mathbb{R}^K$ (the output of a deep neural network with input \mathbf{x}).

Reconstruction Loss. In this paper, data reconstruction of source or target samples is used as an auxiliary task to improve our model’s ability to extract the domain-specific representations. Specifically, we use VAE as the basic reconstruction model, which first encodes a sample \mathbf{x} to a latent representation \mathbf{z} , and then decodes the latent representation back to data space $\tilde{\mathbf{x}}$:

$$\mathbf{z} \sim q(\mathbf{z}|\mathbf{x}) = Ec(\mathbf{x}), \tilde{\mathbf{x}} \sim p(\mathbf{x}|\mathbf{z}) = Dc(\mathbf{z}). \quad (2)$$

The VAE imposes a prior over the latent distribution $p(\mathbf{z})$ to regularize the encoder. Typically $\mathbf{z} \sim N(0, I)$ is chosen. A variational lower bound of the log-likelihood is used as a surrogate objective function:

$$\log p(\mathbf{x}) \geq -D_{KL}[q(\mathbf{z}|\mathbf{x})||p(\mathbf{z})] + \mathbb{E}_{q(\mathbf{z}|\mathbf{x})}[\log p(\mathbf{z}|\mathbf{x})], \quad (3)$$

where D_{KL} is the Kullback-Leibler divergence. Thus, the VAE loss is minus the sum of the expected log likelihood (the reconstruction error) and a prior regularization term:

$$\mathcal{L}_{vae} = \mathcal{L}_{rec} + \mathcal{L}_{prior} = -\mathbb{E}_{q(\mathbf{z}|\mathbf{x})}[\log p(\mathbf{z}|\mathbf{x})] + D_{KL}[q(\mathbf{z}|\mathbf{x})||p(\mathbf{z})]. \quad (4)$$

With the VAE loss, we can not only extract more domain-specific information of each domain, but also allow the structure of source and target domain feature space to be undistorted. Moreover, as the distributions of latent representations form two domains are all forced to be close to a same certain distribution by VAE, the distribution discrepancy between domains might be alleviated to some degree.

Joint Adversarial Loss. As we know, in GAN, the generator G and the discriminator D are trained alternately. Similarly, domain-adversarial learning methods optimize the domain discriminator D with the feature extractor F fixed, and then optimize F with D fixed. Specifically, D and F are optimized alternately according to an adversarial objective, which is shown as follows:

$$\min_D \mathcal{L}_{adv-D}, \min_F \mathcal{L}_{adv-F}. \quad (5)$$

There are various different adversarial loss functions to choose, each of which has their own unique use cases. All adversarial losses train the D using a standard supervised loss:

$$\mathcal{L}_{adv-D} = \mathbb{E}_{\mathbf{x}_s \sim \mathcal{X}_s} \ell_{ce}(D(F(\mathbf{x}_s)), [1, 0]) + \mathbb{E}_{\mathbf{x}_t \sim \mathcal{X}_t} \ell_{ce}(D(F(\mathbf{x}_t)), [0, 1]), \quad (6)$$

where $[1, 0]$ and $[0, 1]$ are the domain labels. However, the loss used to train the F is different. Here, we only take ADDA as an example. It trains the F according to a standard loss function called GAN loss function with inverted labels, which is shown below:

$$\mathcal{L}_{adv-F} = \mathbb{E}_{\mathbf{x}_t \sim \mathcal{X}_t} \ell_{ce}(D(F(\mathbf{x}_t)), [1, 0]). \quad (7)$$

These prior efforts only align the feature distributions across domains at the domain level, but ignore the class-level alignment. To tackle the issue, we introduce a joint domain-class discriminator whose output is a layer with $K + 1$ nodes and employ a joint adversarial loss to train the target model, instead of a simple binary adversarial loss above.

To be specific, we first train a joint domain-class discriminator $D1$ according to a standard supervised loss:

$$\mathcal{L}_{adv-D1} = \mathbb{E}_{(\mathbf{x}_s, y_s) \sim \mathcal{D}_s} \ell_{ce}(D1(Ec_s(\mathbf{x}_s)), d_s^{(1)}) + \mathbb{E}_{\mathbf{x}_t \sim \mathcal{X}_t} \ell_{ce}(D1(Ec_t(\mathbf{x}_t)), d_t^{(1)}). \quad (8)$$

$d_s^{(1)}$ and $d_t^{(1)}$ are defined as follows:

$$d_s^{(1)} = \underbrace{[0, \dots, 0, 1, 0, \dots, 0, 0]}_{y_s}, d_t^{(1)} = \underbrace{[0, \dots, 0, 0, 0, \dots, 0, 1]}_{K}, \quad (9)$$

where y_s is the one-hot, ground-truth source class label. Obviously, the discriminator $D1$ trained by minimizing Eq. 8 cannot distinguish the class of target domain samples, because the first K dimensions of $d_t^{(1)}$ are set to zero. Thereby, we introduce another joint domain-class discriminator $D2$, which is also trained according to a standard supervised loss:

$$\mathcal{L}_{adv-D2} = \mathbb{E}_{(\mathbf{x}_s, y_s) \sim \mathcal{D}_s} \ell_{ce}(D2(Enc_s(\mathbf{x}_s)), d_s^{(2)}) + \mathbb{E}_{\mathbf{x}_t \sim \mathcal{X}_t} \ell_{ce}(D2(Enc_t(\mathbf{x}_t)), d_t^{(2)}). \quad (10)$$

Inversely, $d_s^{(2)}$ and $d_t^{(2)}$ are defined as follows:

$$d_s^{(2)} = \underbrace{[0, \dots, 0, 0, 0, \dots, 0, 1]}_{K}, d_t^{(2)} = \underbrace{[0, \dots, 0, 1, 0, \dots, 0, 0]}_{\hat{y}_t}. \quad (11)$$

where \hat{y}_t is the pseudo-label predicted by the target classifier. In this way, the two joint domain-class discriminators complement each other to align the marginal and class conditional feature distributions of two domains at the same time.

Similar to ADDA above, the target encoder Enc_t is adversarially optimized according to the loss \mathcal{L}_{adv-Ec} with inverted labels to learn domain-invariant representation to fool the joint domain-class discriminators:

$$\mathcal{L}_{adv-Ec} = \mathbb{E}_{\mathbf{x}_t \sim \mathcal{X}_t} \ell_{ce}(D1(Enc_t(\mathbf{x}_t)), [\hat{y}_t, 0]) + \mathbb{E}_{\mathbf{x}_t \sim \mathcal{X}_t} \ell_{ce}(D2(Enc_t(\mathbf{x}_t)), [\mathbf{0}, 1]), \quad (12)$$

where $\mathbf{0}$ is a zero vector of size K . The joint adversarial loss, \mathcal{L}_{adv-D1} , \mathcal{L}_{adv-D2} and \mathcal{L}_{adv-Ec} , contains both domain and class information, so the marginal and class conditional feature distributions of two domains can be aligned.

Entropy Loss. After aligning the feature distributions of two domains, UDA can be regarded as a semi-supervised learning(SSL) problem. The large amount of unlabeled target domain samples can be used to bias the decision boundaries to be in the low-density regions [16]. Applying entropy minimization for classifier training can push the decision boundaries away from the high density regions, a desired property under the cluster assumption [4]. Therefore, our method trains the target classifier C_t according to the following entropy loss:

$$\mathcal{L}_{etp} = \mathbb{E}_{\mathbf{x}_t \sim \mathcal{X}_t} \ell_e(C_t(Enc_t(\mathbf{x}_t))), \quad (13)$$

where $\ell_e(f(\mathbf{x})) := -\langle f(\mathbf{x}), \log f(\mathbf{x}) \rangle$. However, entropy minimization satisfies the cluster assumption only for Lipschitz classifiers [10]. Fortunately, the Lipschitz condition can be realized by virtual adversarial training(VAT) as suggested by [23]. Therefore, as with most previous methods, we apply entropy minimization

in conjunction with VAT on the target domain samples. The VAT loss is given as follows:

$$\mathcal{L}_{vat} = \mathbb{E}_{\mathbf{x}_t \sim \mathcal{X}_t} [\max_{\|r\| \leq \epsilon} D_{KL}(C_t(Ec_t(\mathbf{x}_t)) || C_t(Ec_t(\mathbf{x}_t + r)))]. \quad (14)$$

The above SSL regularizations can be applied after aligning the feature distributions of two domains [34]. But, we find that applying SSL regularizations at the beginning of the training also works well.

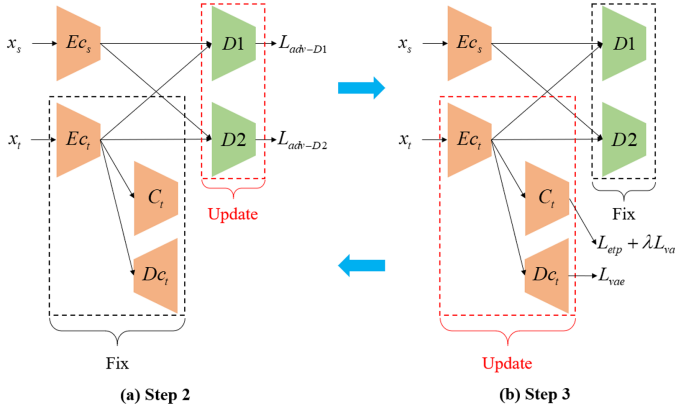


Fig. 3. Adversarial training steps of JDARN

3.4 Training Steps

In this section, we combine the loss functions mentioned above to train all the networks including two VAEs, two joint domain-class discriminators and two classifiers, and summarize the whole training step. Specifically, the training of JDARN is divided into three steps:

Step 1. First of all, the source VAE and source classifier are trained using all the labeled source domain samples according to the classification loss and the VAE loss together. The objective is written as follows:

$$\min_{Ec_s, Dc_s, C_s} \mathcal{L}_{cls} + \lambda_{vae} \mathcal{L}_{vae}, \quad (15)$$

where λ_{vae} is a trade-off parameter. Due to lack of target domain annotations, we use the pre-trained source model as an initialization for the target model, and then fix the source model at the later steps.

Step 2. In this step, the two joint domain-class discriminators are trained according to the joint adversarial loss to discriminate the class and domain of

one latent representation, with the target VAE and classifier fixed. The objective is given as follows:

$$\min_{D_1} \mathcal{L}_{adv-D_1}, \min_{D_2} \mathcal{L}_{adv-D_2}. \quad (16)$$

Step 3. In this step, the target VAE is trained according to the joint adversarial loss to learn the domain-invariant representations to fool the discriminators, with the two discriminators fixed. In addition, VAE loss is added to the objective to help the target VAE extract more domain-specific information. The target classifier is also optimized in this step according to entropy loss and VAT loss, which can push the decision boundaries away from the high density regions. The final objective is given as follows:

$$\min_{E_{c_t}, D_{c_t}, C_t} \mathcal{L}_{adv-Ec} + \lambda_{vae} \mathcal{L}_{vae} + \lambda_t (\mathcal{L}_{etp} + \lambda_{vat} \mathcal{L}_{vat}), \quad (17)$$

where λ_{vae} , λ_t , and λ_{vat} are trade-off parameters.

Step 2 and **Step 3** are conducted in an alternating fashion, as shown in Fig. 3.

3.5 Connection to Domain Adaptation Theory

Most adversarial domain adaptation methods are based on the theoretical analysis proposed in [1, 2], which states that the target error ϵ_t is bounded by four terms:

Theorem 1. *Let \mathcal{H} be a hypothesis space of VC dimension d . If $\mathcal{U}_s, \mathcal{U}_t$ are unlabeled samples of size m each, drawn from P_s and P_t respectively, then for any $\delta \in (0, 1)$, with probability at least $1 - \delta$, for every $h \in \mathcal{H}$,*

$$\begin{aligned} \epsilon_t(h) &\leq \epsilon_s(h) + \frac{1}{2} d_{\mathcal{H}\Delta\mathcal{H}}(P_s, P_t) + \lambda \\ &\leq \epsilon_s(h) + \frac{1}{2} \hat{d}_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{U}_s, \mathcal{U}_t) + 4\sqrt{\frac{2d \log(2m) + \log(\frac{2}{\delta})}{m}} + \lambda, \end{aligned} \quad (18)$$

where $\epsilon_s(h)$ is the expected error on the source domain samples that can be minimized easily with source class label information and $\lambda = \min_h [\epsilon_s(h) + \epsilon_t(h)]$ is combined error of ideal joint hypothesis. $d_{\mathcal{H}\Delta\mathcal{H}}(P_s, P_t)$ denotes the $\mathcal{H}\Delta\mathcal{H}$ -distance between two domains, which is written as follows:

$$d_{\mathcal{H}\Delta\mathcal{H}}(P_s, P_t) = 2 \sup_{h, h' \in \mathcal{H}} |Pr_{x \sim P_s}[h(x) \neq h'(x)] - Pr_{x \sim P_t}[h(x) \neq h'(x)]|. \quad (19)$$

$\hat{d}_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{U}_s, \mathcal{U}_t)$ is the empirical $\mathcal{H}\Delta\mathcal{H}$ -distance.

Let $F_s \# P_s$ and $F_t \# P_t$ be the marginal push-forward distributions of source and target domain distribution P_s and P_t , where F_s and F_t are the feature extractors (encoders in our model). If $F_s \# P_s$ and $F_t \# P_t$ are aligned well, the $d_{\mathcal{H}\Delta\mathcal{H}}$ will vanish for any space \mathcal{H} of sufficiently smooth hypotheses. However,

if the corresponding class conditional distributions $F_s \# P_s(\cdot|y)$ and $F_t \# P_t(\cdot|y)$ are aligned incorrectly, λ will be large because there may not be a $h \in \mathcal{H}$ with low error in both domains. Since our proposed model aligns the marginal and class conditional distributions by the two joint domain-class discriminators and the joint adversarial loss, the above problem can be tackled well. Furthermore, as m increases, the third term will decrease. The m can increase with data augmentation. Therefore, using VAT that has the same effect of augmenting data can make the third term decrease.

4 Experiments

In this section, we evaluate our proposed model with some state-of-the-art deep UDA methods and demonstrate promising results on UDA benchmark tasks.

4.1 Experimental Setup

Office-31 [30] is a standard benchmark dataset for domain adaptation, which consists of 4,110 images spread across 31 classes in 3 domains: Amazon (**A**), Webcam (**W**), and DSLR (**D**). We focus our evaluation on six transfer tasks: **A** \rightarrow **D**, **A** \rightarrow **W**, **D** \rightarrow **A**, **D** \rightarrow **W**, **W** \rightarrow **A** and **W** \rightarrow **D**.

ImageCLEF-DA¹ is a benchmark dataset for ImageCLEF 2014 domain adaptation challenge, which contains three domains: Caltech-256 (**C**), ImageNet ILSVRC 2012 (**I**), and Pascal VOC 2012 (**P**). For each domain, there are 12 classes and 50 images in each class. We evaluate all methods on six transfer tasks: **C** \rightarrow **I**, **C** \rightarrow **P**, **I** \rightarrow **C**, **I** \rightarrow **P**, **P** \rightarrow **C** and **P** \rightarrow **I**.

We compare our proposed model with the following state-of-the-art UDA methods: Deep Adaptation Network (**DAN**) [19], Domain Adversarial Neural Network (**DANN**) [6], Adversarial Discriminative Domain Adaptation (**ADDA**) [37], Virtual Adversarial Domain Adaptation (**VADA**) [34], Maximum Classifier Discrepancy (**MCD**) [32], Conditional Domain Adversarial Network (**CDAN**) [20], and Transferable Adversarial Training (**TAT**) [17]. Besides, as a basic model, ResNet-50 [11] is also used for comparison.

We implement our model in PyTorch [27]². Following the commonly used experimental protocol for UDA [6, 20], we use all labeled source samples and all unlabeled target samples. We repeat each domain adaptation task three times, and then compare the mean classification accuracy. ResNet-50 [11] is adopted as the based model with parameters fine-tuned from the model pre-trained on ImageNet [15].

For our proposed model, the dimension of the latent representation is set to 256; the classifier is a one-layer fully connected network ($256 \rightarrow K$), where K is the number of classes; the discriminator consists of three fully connected layers with ReLU ($256 \rightarrow 3072 \rightarrow 2048 \rightarrow 1024 \rightarrow K + 1$). In all the tasks, the

¹ <https://www.imageclef.org/2014/adaptation>.

² Codes are available at <https://github.com/NaivePawn/JDARN>.

mini-batch stochastic gradient descent (SGD) with momentum of 0.9 is adopted. During pre-training (**Step 1**), batchsize is set to 32; learning rate is set to 10^{-4} for based ResNet-50, and 10^{-3} for the other new layers of source VAE and source classifier that are trained from scratch. During adaptation (**Step 2&3**), batchsize is set to 16; learning rate is set to 10^{-3} for two joint domain-class discriminators, and 10^{-5} for target VAE and classifier.

For hyperparameters λ_{vae} (in Eq. 15 and Eq. 17) and λ_t (in Eq. 17), we fix their value to 0.1. For hyperparameter λ_{vat} (in Eq. 17), we do search over the grid $\{1.0, 10.0\}$, which is suggested by [5]. We also search for the upper bound of the adversarial perturbation in VAT, $\epsilon \in \{0.1, 0.5, 1.0, 2.0, 4.0, 8.0\}$ (in Eq. 14), as in [5]. We train the models within 200 epoches for pre-training and 2000 epoches for adaptation.

Table 1. Classification accuracies (%) on Office-31 for unsupervised domain adaptation with ResNet-50

Method	A→D	A→W	D→A	D→W	W→A	W→D	Avg
RESNET-50 [11]	68.9 ± 0.2	68.4 ± 0.2	62.5 ± 0.3	96.7 ± 0.1	60.7 ± 0.3	99.3 ± 0.1	76.1
DAN [19]	78.6 ± 0.2	80.5 ± 0.4	63.6 ± 0.3	97.1 ± 0.2	62.8 ± 0.2	99.6 ± 0.1	80.4
DANN [6]	81.5 ± 0.4	82.6 ± 0.4	68.4 ± 0.5	96.9 ± 0.2	67.5 ± 0.5	99.3 ± 0.2	82.7
ADDA [37]	77.8 ± 0.3	86.2 ± 0.5	69.5 ± 0.4	96.2 ± 0.3	68.9 ± 0.5	98.4 ± 0.3	82.9
VADA [34]	86.7 ± 0.4	86.5 ± 0.5	70.1 ± 0.4	98.2 ± 0.4	70.5 ± 0.4	99.7 ± 0.2	85.4
MCD [32]	92.2 ± 0.2	88.6 ± 0.2	69.5 ± 0.1	98.5 ± 0.1	69.7 ± 0.3	100.0 ± .0	86.5
CDAN [20]	92.9 ± 0.2	94.1 ± 0.1	71.0 ± 0.3	98.6 ± 0.1	69.3 ± 0.3	100.0 ± .0	87.7
TAT [17]	93.2 ± 0.2	92.5 ± 0.3	73.1 ± 0.3	99.3 ± 0.1	72.1 ± 0.3	100.0 ± .0	88.4
JDARN	93.5 ± 0.2	94.5 ± 0.2	74.2 ± 0.1	98.9 ± 0.1	72.9 ± 0.1	100.0 ± .0	89.0

4.2 Results

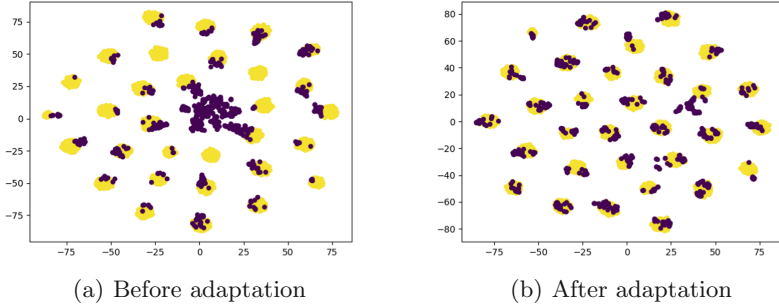
We now discuss the experiment results. The results of baselines are directly reported from the original papers if protocol is the same.

Table 1 shows the results on office-31 dataset based on ResNet-50. The proposed model significantly outperforms all comparison methods on most transfer tasks. It is desirable that our model improves the accuracies on four hard transfer task: **A** → **D**, **A** → **W**, **D** → **A** and **W** → **A**, where the source and target domains are substantially different. Meanwhile, our model produces comparable classification accuracies on easy transfer tasks: **D** → **W** and **W** → **D**, where the source and target domain are similar.

Table 2 shows the results on ImageCLEF-DA dataset based on ResNet-50. Different from Office-31 where different domains are of different sizes, the three domains in ImageCLEF-DA are balanced, which makes it a good complement to Office-31 for more controlled experiments and avoids the class imbalance problems. Our model outperforms all comparison methods on most transfer tasks, especially on tasks **P** → **C** and **P** → **I**.

Table 2. Classification accuracies (%) on Image-CLEF for unsupervised domain adaptation with ResNet-50

Method	C→I	C→P	I→C	I→P	P→C	P→I	Avg
RESNET-50 [11]	78.0 ± 0.2	65.5 ± 0.3	91.5 ± 0.3	74.8 ± 0.3	91.2 ± 0.3	83.9 ± 0.1	80.7
DAN [19]	86.3 ± 0.4	69.2 ± 0.4	92.8 ± 0.2	74.5 ± 0.4	89.8 ± 0.4	82.2 ± 0.2	82.5
DANN [6]	87.0 ± 0.5	74.3 ± 0.5	96.2 ± 0.4	75.0 ± 0.3	91.5 ± 0.6	86.0 ± 0.3	85.0
CDAN [20]	91.3 ± 0.3	74.2 ± 0.2	97.7 ± 0.3	77.7 ± 0.3	94.3 ± 0.3	90.7 ± 0.2	87.7
TAT [17]	92.0 ± 0.4	78.2 ± 0.4	97.5 ± 0.3	78.8 ± 0.2	94.7 ± 0.4	92.0 ± 0.2	88.9
JDARN	92.1 ± 0.1	77.5 ± 0.1	97.0 ± 0.2	79.0 ± 0.2	97.0 ± 0.2	94.0 ± 0.2	89.4

**Fig. 4.** Features visualization. (a) and (b) show the feature representations of two domains before and after the domain adaptation on the task $\mathbf{A} \rightarrow \mathbf{D}$, respectively. Yellow and purple points indicate the source and target domain samples, respectively. After adaptation, target clusters are well-aligned with source clusters. (Color figure online)

4.3 Analysis

Features Visualization. We visualize the feature representations to investigate the effectiveness of domain adaptation on the task $\mathbf{A} \rightarrow \mathbf{D}$ using t-distributed stochastic neighbor embedding (t-SNE) [22]. As shown in Fig. 4(a) and 4(b), the discrepancy between \mathbf{A} and \mathbf{D} is significantly reduced in the latent representation space after adaptation. Moreover, the feature representations with the same class label but from different domains are much more closer after adaptation, which demonstrates the effectiveness of the class-level distribution alignment.

Cross-Domain \mathcal{A} -Distance. The theory of domain adaptation [1] suggests \mathcal{A} -distance as a measure of distribution discrepancy. The proxy \mathcal{A} -distance is defined as $d_{\mathcal{A}} = 2(1 - 2\epsilon)$, where ϵ is the test error of a classifier (e.g. kernel SVM) trained to discriminate the source domain from the target domain. Table 3 shows $d_{\mathcal{A}}$ on tasks $\mathbf{A} \rightarrow \mathbf{W}$, $\mathbf{W} \rightarrow \mathbf{D}$ with features of ResNet-50, DANN, CDAN and JDARN. We observe that $d_{\mathcal{A}}$ on JDARN features is smaller than $d_{\mathcal{A}}$ on both ResNet-50, DANN and CDAN features, which implies that our model can effectively reduce the domain distribution discrepancy. As domains \mathbf{W} and \mathbf{D} are similar, $d_{\mathcal{A}}$ on task $\mathbf{W} \rightarrow \mathbf{D}$ is smaller than that on $\mathbf{A} \rightarrow \mathbf{W}$, implying higher accuracies.

Table 3. Cross-domain \mathcal{A} -distance

Method	A \rightarrow W	W \rightarrow D
RESNET-50 [11]	1.82	1.31
DANN [6]	1.51	1.18
CDAN [20]	1.20	1.10
JDARN	1.12	1.10

Table 4. Ablation experiments

Method	W \rightarrow A	P \rightarrow C
JDARN (w/o vae)	65.4	95.6
JDARN (w binary)	67.2	96.3
JDARN (w/o etp& vat)	71.0	96.7
JDARN	72.9	97.0

Ablation Study. To investigate the effects of each component in our model, we also try ablation study on tasks $\mathbf{W} \rightarrow \mathbf{A}$ and $\mathbf{P} \rightarrow \mathbf{C}$ with different components ablation: (i) removing the decoders and only training the encoders (w/o vae), (ii) training one domain discriminator with a binary adversarial loss (w binary), and (iii) training the target VAE without entropy loss and VAT loss (w/o etp&vat). As seen in Table 4, once one part is removed, the accuracy degrades. The results show that all the losses are designed reasonably and they all promote the classification accuracy for domain adaptation.

5 Conclusion

In this paper, we propose a novel model called Joint Domain-Adversarial Reconstruction Network for unsupervised domain adaptation. Our model mainly tackle two problems of existing domain-adversarial learning methods: ignoring the individual characteristics of each domain and ignoring the class-level alignment. JDARN uses data reconstruction as an auxiliary task to help domain-adversarial learning to learn domain-specific representations. Meanwhile, JDARN uses two joint domain-class discriminators to distinguish both class and domain of the learned representations and trains them according to a joint adversarial loss, so that the distributions of two domains can be aligned at the domain level and the class level simultaneously. Comprehensive experiments on two standard UDA datasets and tasks verify the effectiveness of our proposed model.

Acknowledgements. This paper is supported by the National Key Research and Development Program of China (Grant No. 2018YFB1403400), the National Natural Science Foundation of China (Grant No. 61876080), the Key Research and Development Program of Jiangsu (Grant No. BE2019105), the Collaborative Innovation Center of Novel Software Technology and Industrialization at Nanjing University.

References

1. Ben-David, S., Blitzer, J., Crammer, K., Kulesza, A., Pereira, F., Vaughan, J.W.: A theory of learning from different domains. *Mach. Learn.* 151–175 (2009). <https://doi.org/10.1007/s10994-009-5152-4>

2. Ben-David, S., Blitzer, J., Crammer, K., Pereira, F.: Analysis of representations for domain adaptation. In: *Advances in Neural Information Processing Systems*, pp. 137–144 (2007)
3. Bousmalis, K., Trigeorgis, G., Silberman, N., Krishnan, D., Erhan, D.: Domain separation networks. In: *Advances in Neural Information Processing Systems*, pp. 343–351 (2016)
4. Chapelle, O., Zien, A.: Semi-supervised classification by low density separation. In: *AISTATS*, vol. 2005, pp. 57–64. Citeseer (2005)
5. Cicek, S., Soatto, S.: Unsupervised domain adaptation via regularized conditional alignment. arXiv preprint [arXiv:1905.10885](https://arxiv.org/abs/1905.10885) (2019)
6. Ganin, Y., Lempitsky, V.: Unsupervised domain adaptation by backpropagation (2014)
7. Ganin, Y., et al.: Domain-adversarial training of neural networks. *J. Mach. Learn. Res.* **17**(1), 2096–2130 (2016)
8. Ghifary, M., Kleijn, W.B., Zhang, M., Balduzzi, D., Li, W.: Deep reconstruction-classification networks for unsupervised domain adaptation. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) *ECCV 2016*. LNCS, vol. 9908, pp. 597–613. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46493-0_36
9. Goodfellow, I., et al.: Generative adversarial nets. In: *Advances in Neural Information Processing Systems*, pp. 2672–2680 (2014)
10. Grandvalet, Y., Bengio, Y.: Semi-supervised learning by entropy minimization. In: *Advances in Neural Information Processing Systems*, pp. 529–536 (2005)
11. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778 (2016)
12. Hu, L., Kan, M., Shan, S., Chen, X.: Duplex generative adversarial network for unsupervised domain adaptation. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1498–1507 (2018)
13. Kang, G., Zheng, L., Yan, Y., Yang, Y.: Deep adversarial attention alignment for unsupervised domain adaptation: the benefit of target expectation maximization. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) *ECCV 2018*. LNCS, vol. 11215, pp. 420–436. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01252-6_25
14. Kingma, D.P., Welling, M.: Auto-encoding variational bayes (2013)
15. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: *Advances in Neural Information Processing Systems*, pp. 1097–1105 (2012)
16. Kumar, A., et al.: Co-regularized alignment for unsupervised domain adaptation. In: *Advances in Neural Information Processing Systems*, pp. 9345–9356 (2018)
17. Liu, H., Long, M., Wang, J., Jordan, M.: Transferable adversarial training: a general approach to adapting deep classifiers. In: *International Conference on Machine Learning*, pp. 4013–4022 (2019)
18. Liu, M.Y., Tuzel, O.: Coupled generative adversarial networks. In: *Advances in Neural Information Processing Systems*, pp. 469–477 (2016)
19. Long, M., Cao, Y., Wang, J., Jordan, M.I.: Learning transferable features with deep adaptation networks (2015)
20. Long, M., Cao, Z., Wang, J., Jordan, M.I.: Conditional adversarial domain adaptation. In: *Advances in Neural Information Processing Systems*, pp. 1640–1650 (2018)

21. Long, M., Zhu, H., Wang, J., Jordan, M.I.: Deep transfer learning with joint adaptation networks. In: Proceedings of the 34th International Conference on Machine Learning, vol. 70, pp. 2208–2217. JMLR. org (2017)
22. Maaten, L.V.D., Hinton, G.: Visualizing data using t-SNE. *J. Mach. Learn. Res.* **9**(Nov), 2579–2605 (2008)
23. Miyato, T., Maeda, S.I., Koyama, M., Ishii, S.: Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *IEEE Trans. Pattern Anal. Mach. Intell.* **41**(8), 1979–1993 (2018)
24. Motiian, S., Piccirilli, M., Adjeroh, D.A., Doretto, G.: Unified deep supervised domain adaptation and generalization. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 5715–5725 (2017)
25. Pan, S.J., Tsang, I.W., Kwok, J.T., Yang, Q.: Domain adaptation via transfer component analysis. *IEEE Trans. Neural Netw.* **22**(2), 199–210 (2010)
26. Pan, S.J., Yang, Q.: A survey on transfer learning. *IEEE Trans. Knowl. Data Eng.* **22**(10), 1345–1359 (2009)
27. Paszke, A., et al.: PyTorch: an imperative style, high-performance deep learning library. In: Advances in Neural Information Processing Systems, pp. 8024–8035 (2019)
28. Pei, Z., Cao, Z., Long, M., Wang, J.: Multi-adversarial domain adaptation. In: Thirty-Second AAAI Conference on Artificial Intelligence (2018)
29. Quiñero-Candela, J., Sugiyama, M., Schwaighofer, A., Lawrence, N.: Covariate shift and local learning by distribution matching (2008)
30. Saenko, K., Kulis, B., Fritz, M., Darrell, T.: Adapting visual category models to new domains. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010. LNCS, vol. 6314, pp. 213–226. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-15561-1_16
31. Saito, K., Ushiku, Y., Harada, T.: Asymmetric tri-training for unsupervised domain adaptation. In: Proceedings of the 34th International Conference on Machine Learning, vol. 70, pp. 2988–2997. JMLR. org (2017)
32. Saito, K., Watanabe, K., Ushiku, Y., Harada, T.: Maximum classifier discrepancy for unsupervised domain adaptation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3723–3732 (2018)
33. Sener, O., Song, H.O., Saxena, A., Savarese, S.: Learning transferrable representations for unsupervised domain adaptation. In: Advances in Neural Information Processing Systems, pp. 2110–2118 (2016)
34. Shu, R., Bui, H.H., Narui, H., Ermon, S.: A DIRT-T approach to unsupervised domain adaptation (2018)
35. Torralba, A., Efros, A.A., et al.: Unbiased look at dataset bias. In: CVPR, vol. 1, p. 7. Citeseer (2011)
36. Tzeng, E., Hoffman, J., Darrell, T., Saenko, K.: Simultaneous deep transfer across domains and tasks. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 4068–4076 (2015)
37. Tzeng, E., Hoffman, J., Saenko, K., Darrell, T.: Adversarial discriminative domain adaptation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 7167–7176 (2017)
38. Wang, M., Deng, W.: Deep visual domain adaptation: a survey. *Neurocomputing* **312**, 135–153 (2018)

39. Xie, S., Zheng, Z., Chen, L., Chen, C.: Learning semantic representations for unsupervised domain adaptation. In: International Conference on Machine Learning, pp. 5419–5428 (2018)
40. Zhang, W., Ouyang, W., Li, W., Xu, D.: Collaborative and adversarial network for unsupervised domain adaptation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3801–3809 (2018)