

Deep Ladder-Suppression Network for Unsupervised Domain Adaptation

Wanxia Deng¹, Lingjun Zhao, Gangyao Kuang, *Senior Member, IEEE*, Dewen Hu², *Senior Member, IEEE*,
Matti Pietikäinen³, *Fellow, IEEE*, and Li Liu⁴, *Senior Member, IEEE*

Abstract—Unsupervised domain adaptation (UDA) aims at learning a classifier for an unlabeled target domain by transferring knowledge from a labeled source domain with a related but different distribution. Most existing approaches learn domain-invariant features by adapting the entire information of the images. However, forcing adaptation of domain-specific variations undermines the effectiveness of the learned features. To address this problem, we propose a novel, yet elegant module, called the deep ladder-suppression network (DLSN), which is designed to better learn the cross-domain shared content by suppressing domain-specific variations. Our proposed DLSN is an autoencoder with lateral connections from the encoder to the decoder. By this design, the domain-specific details, which are only necessary for reconstructing the unlabeled target data, are directly fed to the decoder to complete the reconstruction task, relieving the pressure of learning domain-specific variations at the later layers of the shared encoder. As a result, DLSN allows the shared encoder to focus on learning cross-domain shared content and ignores the domain-specific variations. Notably, the proposed DLSN can be used as a standard module to be integrated with various existing UDA frameworks to further boost performance. Without whistles and bells, extensive experimental results on four gold-standard domain adaptation datasets, for example: 1) Digits; 2) Office31; 3) Office-Home; and 4) VisDA-C, demonstrate that the proposed DLSN can consistently and significantly improve the performance of various popular UDA frameworks.

Index Terms—Autoencoder, deep convolutional neural network (DCNN), deep learning, domain adaptation (DA), transfer learning, unsupervised learning.

Manuscript received 8 December 2020; accepted 4 March 2021. Date of publication 30 March 2021; date of current version 16 September 2022. This work was supported in part by the National Natural Science Foundation of China under Grant 9186301 and Grant 61701508; in part by the Hunan Science and Technology Plan Project under Grant 2019GK2131; in part by the Hunan Provincial Natural Science Foundation of China under Grant 2018JJ3613; in part by Academy of Finland under Grant 331883; and in part by the National Natural Science Foundation of China under Grant 61872379. This article was recommended by Associate Editor Y. Zhang. (Corresponding author: Li Liu.)

Wanxia Deng, Lingjun Zhao, and Gangyao Kuang are with the State Key Laboratory of Complex Electromagnetic Environment Effects on Electronics and Information System, College of Electronic Science, National University of Defense Technology, Changsha 410073, China (e-mail: wanxiadeng@163.com; nudtzhj@163.com; kuanggangyao@nudt.edu.cn).

Dewen Hu is with the College of Intelligent Science, National University of Defense Technology, Changsha, China (e-mail: dwhu@nudt.edu.cn).

Matti Pietikäinen is with the Center for Machine Vision and Signal analysis, University of Oulu, 90570 Oulu, Finland (e-mail: matti.pietikainen@oulu.fi).

Li Liu is with the College of System Engineering, National University of Defense Technology, Changsha 410073, China, and also with the Center for Machine Vision and Signal analysis, University of Oulu, 90570 Oulu, Finland (e-mail: li.liu@oulu.fi).

Color versions of one or more figures in this article are available at <https://doi.org/10.1109/TCYB.2021.3065247>.

Digital Object Identifier 10.1109/TCYB.2021.3065247

I. INTRODUCTION

SINCE 2012, when the field of artificial neural networks was reignited by the breakthrough ImageNet classification result obtained by a deep convolutional neural network (DCNN), called AlexNet [1], deep neural networks (DNNs) have shone in a broad range of areas in computer vision, such as image classification [2]; object detection [3]; image segmentation [4], [5]; face recognition [6] and many others. Nevertheless, the impressive progress depends on strong supervision, that is, massive amounts of annotated data that are painstakingly labeled by numerous workers or specialists. Labeling instances is difficult, expensive, or even time consuming because it requires experienced human annotators' efforts. In some applications, such as privacy-sensitive applications and the medical domain, labeled samples are even impossible. These days, unlabeled data may be relatively easy to collect, but effective ways to exploit them are lacking.

To circumvent the issues mentioned above, an alternative approach (e.g., transfer learning) is to leverage labeled data and/or models from a similar domain (dubbed *source domain*), where large-scale labeled training data (e.g., synthetic data) can be more easily obtained, to improve the model for the domain of interest (dubbed *target domain*). This is because transfer learning aims to decrease the human effort and obtain high accuracy, and has received significant attention in recent years. However, the current evidence [7] indicates that DNNs have a strong reliance on the dataset with which they are originally trained. The learned features cannot be easily transferred to a different but relevant domain without adjusting. Analogous to other machine-learning techniques, DNNs also suffer from the problem of domain shift [8]–[10], that is, predictors trained on a dataset undergo a drastic drop in performance when applied to the target domain. The domain shift refers to the difference in data distributions between source and target domains, which is caused by many changing conditions, e.g., background and camera poses, occlusions, volatile illumination conditions, etc. To mitigate this degradation, domain adaptation (DA) has been proposed to address the problem of the domain shift between the source and target domains [11], [12].

In this article, we focus on the closed set, the unsupervised DA (UDA), where the source domain and the target domain have the same classification task. The main objective is to transfer knowledge from a labeled source domain to a fully unlabeled target domain with a related but different data distribution. Recently, DNNs have injected new vitality into the field of UDA, and significant progress has been witnessed

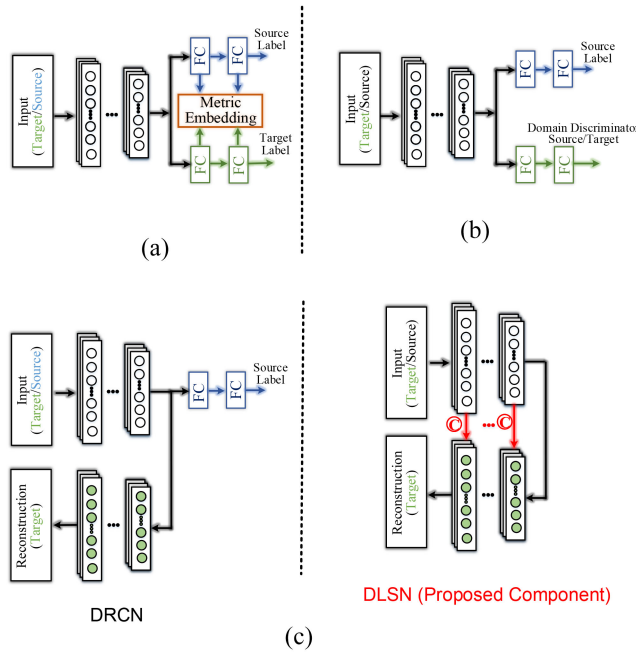


Fig. 1. Contrasting the proposed DLSN with state-of-the-art methods. Our proposed DLSN is designed to better learn the cross-domain shared content by suppressing domain-specific variations with a ladder network. The color of “Source” and “Target” denotes that source and target data go by the corresponding branch. The black of “Source” indicates that the source data will be passed into these two branches. See the texts in the introduction for details. (a) Metric discrepancy-based methods. (b) Adversarial learning-based methods. (c) Reconstruction-based methods.

in this field [13]–[16]. Despite the progress, most deep UDA approaches have not been, however, capable of performing at a level sufficient for real-world applications in terms of both accuracy and computational efficiency [15], [16].

Theoretical studies [17], [18] have shown that learning proper cross-domain representations is an efficient path for addressing the domain shift. Therefore, inspired by this, most existing UDA approaches are devoted to using the prior information of two domains to extract the domain-invariant features while keeping discriminative for the main learning task for the source domain during the learning. The mainstream approaches can be roughly grouped into two main categories as follows.

- 1) *Metric discrepancy-based methods* [19]–[26] aim at minimizing the discrepancy between the feature distributions of two domains, which is shown in Fig. 1(a).
- 2) *Adversarial learning-based methods* [27]–[33] introduce a domain discriminator to discriminate the source and the target domains and then borrow adversarial ideas from the generative adversarial network (GAN) [34] to deceive the domain discriminator to obtain indistinguishable features. The framework of the type of methods is in Fig. 1(b).

There are two drawbacks in existing methods based on the metric discrepancy and adversarial learning. One drawback is that these approaches straightforwardly apply the target domain to the adaptation of distributions without altogether mining the target domain’s structure information. Although the target domain is unlabeled, the latent information can

still be mined and helps explore domain-invariant features. The other drawback is that these methods mainly align the global images across the source and target domains. However, it is evident that not all image regions are transferable, while forcefully aligning the domain-specific variations, e.g., images backgrounds and object textures, may lead to negative transfer. The adaptation should focus on high-level abstract features.

Aiming at solving the first drawback, one representative method, deep reconstruction-classification networks (DRCNs), [35] explores the structure of the target domain using unsupervised reconstruction. The DRCN depicted in Fig. 1(c), which is more related to our proposed approach, intends to achieve adaptation of two domains by jointly training a shared encoder for two tasks: 1) the supervised classification task on the source data, which aims at learning discriminative task-specific features and 2) the unsupervised reconstruction task on the unlabeled target data, where the objective of the autoencoder is preserving the complete information (all details) needed for pixel-level image reconstruction task. In this way, the learned representation can encode structural information of the target domain and improve adaptation performance. However, DRCN is also faced with the second drawback, that is, applying all features of an image to adapt. Too many unnecessary features can ultimately lead to difficulties in performing the adaptation and will impair the adaptation performance. To concurrently overcome the two drawbacks, we propose a novel module, called the deep ladder-suppression network (DLSN), for UDA. The proposed DLSN is shown in Fig. 1(c) compared to popular UDA architectures. Compared to the DRCN, the DLSN not only explores the structure information of the target domain, but can also suppress the unnecessary feature to make higher layers of the encoder capture high-level abstract representations via the skipping connections.

The presented DLSN is motivated by the recent success of the ladder network for traditional semisupervised learning [36]–[38]. The ladder network is an autoencoder with skipping connections from the encoder to the decoder, which is demonstrated that the skip connections can relieve the pressure to represent details in the higher layers of the model, because, through the skip connections, the decoder can recover any details discarded by the encoder [37]. Motivated by the insight, we introduce the ladder network for the UDA, which not only mines the information of the target domain but also guarantees to suppress the domain-specific variations and obtains the cross-domain shared representation. The proposed DLSN consists of an encoder with cross-domain sharing and a target domain reconstruction decoder. The encoder and decoder layers are connected by lateral shortcuts. The lateral shortcuts, that is, ladder layers [39], [40], are *crucial*, which can alleviate the pressure to learn domain-specific variations at the later layers of the shared encoder, because the domain-specific components are directly fed to the reconstruction decoder. By this means, the shared encoder can focus on learning cross-domain shared representations by suppressing the domain-specific variations.

The proposed DLSN can readily be used as a module to be flexibly integrated with state-of-the-art UDA frameworks (e.g.,

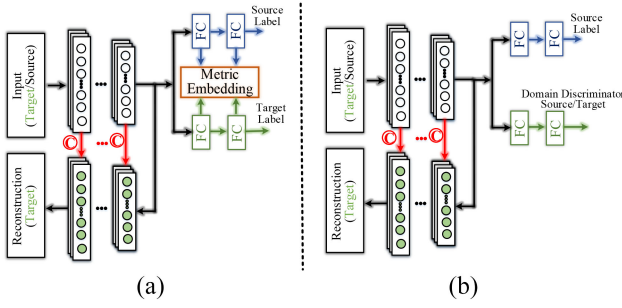


Fig. 2. Proposed DLSN is integrated into the (a) metric discrepancy-based methods and (b) adversarial learning-based methods, respectively.

metric discrepancy-based methods and adversarial learning-based methods to boost performance, and can be trained end to end. An existing UDA approach called “X” integrated with our proposed DLSN module will be referred to as “X + DLSN,” which is as shown in Fig. 2. As a case study, we plug our proposed DLSN into several different representative network architectures e.g., deep CORAL (D-CORAL) [24], deep adaptation network (DAN) [19] and joint adaptation network (JAN) [21] from metric discrepancy-based methods and domain-adversarial neural network (DANN) [29] and conditional domain adversarial network (CDAN) [33] from adversarial learning-based methods.

UDA frameworks integrated with our proposed DLSN can be trained by jointly learning three tasks: 1) supervised classification on the source data; 2) unsupervised reconstruction of the target data; and 3) adapting cross-domain shared representation. In summary, the main contributions of this article are listed as follows.

- 1) By first introducing the ladder autoencoder network into UDA, we propose a novel module DLSN, which can learn expressive cross-domain shared content by suppressing domain-specific variations. The lateral connections in the autoencoder are playing a vital role in boosting UDA performance.
- 2) The proposed DLSN can be readily used as a module to be flexibly integrated into many existing UDA frameworks (e.g., metric discrepancy-based methods and adversarial learning-based methods) to significantly boost their performance.
- 3) Extensive experiments on the Digit, Office31, Office-Home, and VisDA-C datasets are provided to demonstrate the effectiveness of the proposed DLSN and show that DLSN gives a significant boost in performance.

The remainder of this article is organized as follows. In Section II, we review the related work. Section III introduces the formulation of our network. Section IV gives experimental results and analysis on Digit, Office-31, Office-Home, and VisDA-C datasets. Our conclusion is presented in Section V.

II. RELATED WORKS

Research in UDA is so vast that only closely related work is discussed in this section and organized in different categories.

Metric Discrepancy-Based Methods: One common approach for UDA is to guide the domain-invariant feature

learning by minimizing the domain distribution discrepancy with the metric paradigm. Some representative metric methods include maximum mean discrepancy (MMD) [19]–[21], [23], [25], [26], [41]–[43]; CORrelation (CORAL) alignment [44]; and central moment discrepancy (CMD) [22]. The DDC [41] utilizes the MMD metric in the last fully connected layer in addition to the regular classification loss to learn representations that are both domain-invariant and discriminative. In [19] and [23], the DAN is proposed to enhance the feature transferability by minimizing a multikernel MMD in several task-specific layers. The JAN [21] learns a transfer network by aligning the joint distributions of multiple domain-specific layers across domains based on a joint MMD (JMMD) criterion. The paper [42] proposes a weighted MMD model, which introduces class-specific auxiliary weights into the original MMD for exploiting the class prior probability on the source and target domains. The contrastive adaptation network (CAN) [43] explicitly models the intraclass-domain discrepancy and the interclass-domain discrepancy based on the MMD metric. The D-CORAL [24] extends CORAL to learn a nonlinear transformation that aligns correlations of layer activations in (DNNs) by adding the adaptation layer. The CMD [22] proposes to match the higher-order central moments of probability distributions employing order-wise moment differences.

Adversarial Learning-Based Methods: More successful approaches toward DA are based on the adversarial learning of features. Adversarial learning has been widely applied in the DA to deceive the domain discriminator. RevGrad [27], DANN [29], adversarial discriminative DA (ADDA) [28], and CDAN [33] add a subnetwork as the domain discriminator. The extracted feature is learned to fool the domain discriminator using a domain-adversarial learning technique. Many adversarial DA methods are extensions of the representative DANN, such as moving semantic transfer network (MSTN) [45], transferable adversarial training (TAT) [46], and so on. MADA [32] captures multimode structures to enable the fine-grained alignment of different data distributions based on multiple domain discriminators. The Wasserstein distance-guided representation learning (WDGRL) [30] estimates the empirical Wasserstein distance between the source and target samples in a domain-critic network and optimizes the feature extractor network to minimize the estimated Wasserstein distance in an adversarial manner. Similar to the motivation of WDGRL, the re-weighted adversarial adaptation network (RAAN) [47] minimizes the optimal transport (OT)-based Earth-mover (EM) distance and reformulates it to a minimax objective function. Unlike the above domain adversarial learning, maximum classifier discrepancy (MCD) [48] defines a new adversarial standard in developing generic DA frameworks. The MCD utilizes task-specific classifiers as discriminators and aligns distributions of the source and target by the adversarial learning of two task-specific classifiers. Similar to the adaptation standard of the MCD, sliced Wasserstein discrepancy (SWD) [49] adopts the Wasserstein metric to minimize the cost of moving the marginal distributions between the task-specific classifiers.

Generative Adversarial Networks-Based Methods: There are more popular DA approaches that incorporate the generative model into the feature learning process using GAN. The pixel-level DA (PixelDA) [50] learns to generate a new version of the source images with the style of the target domain so that one shared classifier can accommodate both domains. Couple GAN (CoGAN) [51] trains coupled generators and discriminators by weight-sharing to learn the joint distribution across the two domains. The domain-transfer network (DTN) [52] synthesizes source-domain samples that resemble target domain ones by the extra consistency constraint that the same asymmetric transformation should keep the target domain samples identical. Generate to adapt (GTA) [53] proposes an adversarial image generation approach to learn the feature embedding using a combination of generated source-like image classification loss and an image generation procedure. The cycle-consistent adversarial DA (CyCADA) [54], the unsupervised image-to-image translation (UNIT) [55], deep adversarial attention alignment (DAAA) [56], and SBADA-GAN [57] constrain the mapping to be well covered across two domains by imposing cycle consistency: the mapping in one direction (source-to-target or target-to-source) should get back where it started. The image to image translation (I2I) [58] combines domain-agnostic feature extraction, domain generation, and cycle consistency into a single unified framework for domain knowledge.

Autoencoder Reconstruction-Based Methods: The DA approaches based on autoencoder reconstruction learn the domain-invariant feature with a shared encoder and a reconstruction loss of the autoencoder. DRCN [35] is proposed to learn the common encoding representation by combining the supervised classification of the source domain and unsupervised reconstruction of the target domain. The domain separation network (DSN) [59] proposes to separate the feature into the shared feature and the private feature. These two features are encouraged to be orthogonal while both features can be decoded back to images. The shared part is adopted for classification. On the basis of the DSN, the work [60] proposes that orthogonal regularization is applied between private elements across domains, in addition to private features and shared features in each domain.

In summary, the proposed DLSN mainly belongs to autoencoder reconstruction-based methods. Our DLSN leverages labeled source data and unlabeled target data for learning informative domain-invariant features. The novelty is on the exploitation of aligning informative features and filtering out uninformative features by the ladder network, which has not been explored in the UDA literature.

III. PROPOSED METHODOLOGY

In this section, we first give the UDA problem formulation and then introduce the proposed DLSN. Then, we show how to adapt the cross-domain shared features, which mainly describe the combination of existing UDA methods and the proposed DLSN. Finally, the way of the network training is depicted.

A. UDA Problem Formulation

Consider an input space (or a feature representation space) $\mathcal{X}$ and an output space (or a label space) $\mathcal{Y}$. In vanilla supervised learning, we are provided with (input, label) data pairs $\{(\mathbf{x}_i, y_i)\}_i^N$, which are drawn *independent identically distributed* from a distribution $p(\mathbf{x}, y)$ over the input-label space pair $\mathcal{X} \times \mathcal{Y}$. The objective of a machine-learning algorithm is to learn a model (a hypothesis) $h : \mathcal{X} \rightarrow \mathcal{Y}$, which is able to map an input example from $\mathcal{X}$ to an output label from $\mathcal{Y}$ and more importantly can predict labels for new examples coming from the same distribution with the least error. In the context of DA, we consider two different domains defined with different but related probability distributions $p(\mathbf{x}, y)$ over the input-label space pair $\mathcal{X} \times \mathcal{Y}$. Specifically, the domain of interest is dubbed the target domain with the distribution $p_t(\mathbf{x}, y)$ and the domain with labeled data is called the source domain with the distribution $p_s(\mathbf{x}, y)$. The closed set UDA goal, where the source domain and the target domain have the same classification task, is to predict the labels of samples drawn from a target domain as accurately as possible, given N_s labeled samples $D_s = \{(\mathbf{x}_i^s, y_i^s)\}_{i=1}^{N_s}$ drawn from a source domain and N_t unlabeled samples $D_t = \{(\mathbf{x}_i^t)\}_{i=1}^{N_t}$ sampled from the target domain itself. Our goal is to learn feature transformations to map the input space $\mathcal{X}$ to a feature space $\mathcal{F}$, where distributions of the two domains are well aligned in order to obtain the domain-invariant feature so that the target domain images can be classified into $\mathcal{Y}$ correctly.

B. DLSN: Proposed Deep Ladder-Suppression Network

Since the target data are unlabeled, how to explore the supervisory signal to mine the information underlying the unlabeled data is critical. Reconstruction of input samples from low-dimensional latent representations at multilayers of the DCNN is a general way of learning important representation in an unsupervised fashion, which has been explored for UDA such as the recent work DRCN [35] and DSN [59].

In UDA, although the target domain is unlabeled, the structure information is useful and worth exploring. Based on this, the DRCN [35] applies the reconstruction of the target domain to mine the structure features while conducting supervised classification of the labeled source data. In this way, the learned representation preserves discriminability and encodes structure information of the target domain. In this way, the learned representation preserves discriminability and encodes structure information of the target domain. However, DRCN still applies all the details of images to adapt. The uninformative information will impair the performance of the adaptation.

In this work, inspired by the recent success of the ladder network [36]–[38] for traditional semisupervised learning, we propose a novel module dubbed DLSN that can promote the emergence of cross-domain shared representations by suppressing the domain-specific variations. The proposed DLSN explores the reconstruction task as the supervisory signal to mine the important information underlying the unlabeled target data with a ladder network [36]–[38], which is a reconstruction autoencoder with lateral connections from the encoder

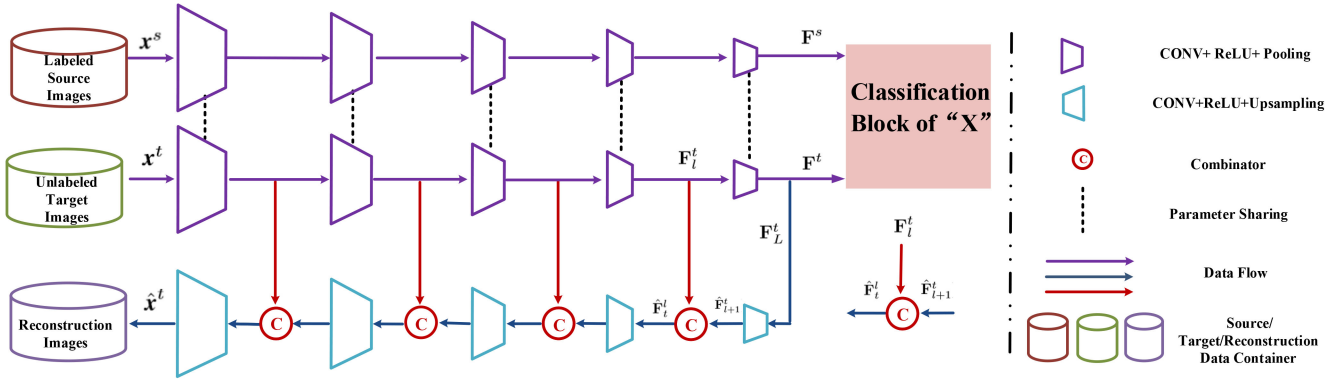


Fig. 3. Illustration of the proposed DLSN. The DLSN is constructed with a ladder autoencoder. The encoder is shared by the source images and target images. The solid red lines denote the shortcuts. The domain-specific components are merged into the decoder of the DLSN by lateral shortcuts for reconstruction.

to the decoder. The learning task is similar to that in vanilla reconstruction autoencoders but applied at each layer, not just the input layer. The skip connections play a vital role in this architecture. They relieve the pressure to represent details at the later layers of the encoder because low-level details from earlier layers of the encoder are directly fed to the decoder through the lateral connections. By this means, the shared encoder, that is, the feature extractor, can focus on learning domain-invariant representations for the classification task on the source data, and in the meantime, the decoder can recover details discarded by the encoder. The overall architecture of our proposed DLSN is illustrated in Fig. 3. We introduce the ladder network, that is, connecting the shared encoder and the decoder layers with lateral shortcuts (the red connections shown in Fig. 3), which alleviate the pressure to learn details at the later layers of the shared encoder. The encoder is shared by the two domains. The autoencoder with ladder connections is applied to mine the target domain's representation and suppress the domain-specific variations.

Formally, our proposed DLSN can be formulated as follows. We define the shared encoder of DLSN as G^{en} with parameters θ^{en} , the decoder as G^{de} with parameters θ^{de} , and the classification block on the source data as G^{cls} with parameters θ^{cls} . G_l^{en} consists of the convolutional, ReLU, or downsampling layers. G_l^{de} consists of the convolutional, ReLU, or upsampling layers. As defined in Section III-A, we are given N_s labeled samples $D_s = \{(x_i^s, y_i^s)\}_{i=1}^{N_s}$ drawn from a source domain and N_t unlabeled samples $D_t = \{(x_i^t)\}_{i=1}^{N_t}$ sampled from the target domain. For the target domain, the forward propagation of each layer is described more formally as follows. An input image x^t from the target domain undergoes the encoder to generate hierarchical feature representations

$$\mathbf{F}_l^t = \begin{cases} G_l^{\text{en}}(x^t), & l = 1 \\ G_l^{\text{en}}(\mathbf{F}_{l-1}^t), & 2 \leq l \leq L \end{cases} \quad (1)$$

where L represents the total number of layers of the encoder, $\mathbf{F}_l^t \in \mathbb{R}^{H_l \times W_l \times C_l}$, $H_l \times W_l$ denotes the size of the feature map at layer l , and C_l is the feature channels at layer l . The feature $\mathbf{F}_L^t$ from the encoder undergoes the decoder with lateral connections to reconstruct $\hat{x}^t$ via the following:

$$\hat{\mathbf{F}}_l^t = \begin{cases} G_l^{\text{de}}(\mathbf{F}_l^t), & l = L \\ G_l^{\text{de}}(\text{Combinator}(\mathbf{F}_l^t, \hat{\mathbf{F}}_{l+1}^t)), & L > l \geq 1 \end{cases} \quad (2)$$

where $\hat{\mathbf{F}}_l^t$ indicates the l th layer reconstruction feature of the target domain in the decoder. $\hat{\mathbf{F}}_1^t$ is the reconstructed output $\hat{x}^t$. $\hat{x}^t$ is the output of the decoder, that is, it is the reconstruction of x^t . The combinator in (2) is defined as follows:

$$\text{Combinator}(\mathbf{F}_l^t, \hat{\mathbf{F}}_{l+1}^t) = \mathbf{Z} \odot \mathbf{F}_l^t + (\mathbf{A} - \mathbf{Z}) \odot \hat{\mathbf{F}}_{l+1}^t \quad (3)$$

where $\odot$ is the elementwise multiplication operator, $\mathbf{A}$ is a 3-D matrix with all 1 elements, and $\mathbf{Z}$ is defined as

$$\mathbf{Z} = \text{Sigmoid}(\mathbf{P}_1 \odot \mathbf{F}_l^t + \mathbf{P}_2) \quad (4)$$

where $\mathbf{P}_1$ and $\mathbf{P}_2$ are trainable parameters, whose elements are initialized as 1. $\mathbf{P}_1, \mathbf{P}_2 \in \mathbb{R}^{H_l \times W_l \times C_l}$, $H_l \times W_l$ denotes the size of the feature map at layer l , and C_l is the feature channels at layer l . The nonlinear combination is a connection, which plays an important role in enabling the proposed DLSN to focus on learning cross-domain shared features while suppressing domain-specific variations. Next, we will present in great detail how to integrate the proposed DLSN module with the existing UDA frameworks, that is, “X + DLSN.”

C. X + DLSN: Integrating DLSN With Existing UDA Frameworks

In this section, we introduce how to integrate the DLSN into the existing UDA algorithms to enable cross-domain shared feature adaptation. The source-domain classification is carried out via the shared encoder and classification layer. Meanwhile, the adaptation of cross-domain shared features is conducted via explicitly (e.g., metric discrepancy-based methods) or implicitly (e.g., adversarial learning-based methods) minimizing the domain distribution discrepancy, which is illustrated in Fig. 4. As shown in Fig. 4(a) and (b), the forward propagation of the DLSN block and label prediction branch is the same, which proceeds as follows:

$$\mathbf{F}^{s/t} = G_{\text{en}}(x^{s/t}) \quad (5)$$

$$\mathbf{H}_v^{s/t} = G_v^{\text{cls}}(\mathbf{F}^{s/t}), \quad \text{for } 1 \leq v \leq V \quad (6)$$

$$\tilde{y}^s = \text{Softmax}(\mathbf{H}_V^s) \quad (7)$$

where $\mathbf{F}^{s/t}$ denotes the representation of the source or target domains in the encoder's final layer. Here, V represents the total number of layers of the classification block. v refers

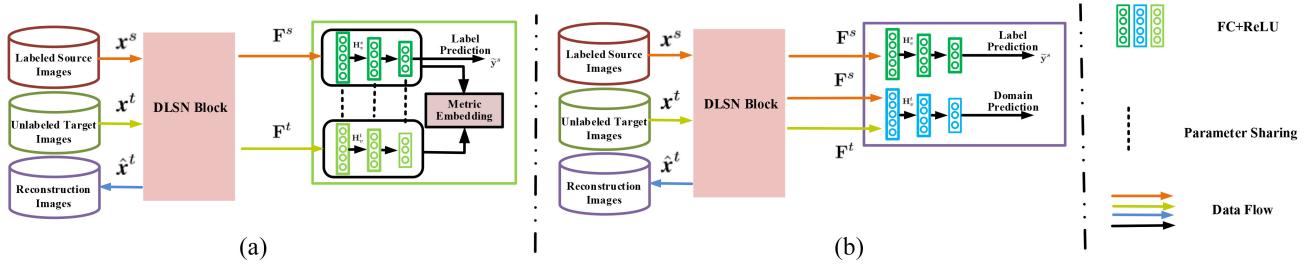


Fig. 4. Illustration of the proposed approach. The proposed DLSN block as a component can be combined with (a) metric discrepancy-based methods and (b) adversarial learning-based methods, respectively. The training phase consists of two parallel pipelines: 1) classification branch where the shared encoder of the DLSN block and label predictor block are updated with the classification loss of the source domain and the discrepancy loss of cross-domain shared features (when the DLSN is integrated into the Metric discrepancy-based methods, such as D-CORAL [24], DAN [19], JAN [21], and so on), or the domain adversarial loss (when the DLSN is integrated into the adversarial learning-based methods, such as DANN [29] and CDAN [33]) and 2) data reconstruction branch that consists of a ladder autoencoder. The cross-domain shared features are propagated through the shared encoder of the DLSN for classification, and the domain-specific components are merged into the decoder of the DLSN by lateral shortcuts for reconstruction. Notably, when the DLSN block is integrated with metric discrepancy-based methods (metric discrepancy block) or Adversarial learning-based methods (adversarial learning block), lines of the same color represent the same data flow.

to variables on the classification layers. G_v^{cls} denotes the feature transformation on the v th layer of the classification block. $\mathbf{H}_v^{s/t}$ is the representation of source or target domains on the v th layer of the classification block. $\hat{\mathbf{y}}^s$ denotes the network's prediction of $\mathbf{x}^s$.

1) *Integrated With Metric Discrepancy-Based Methods:* There are multiple choices for the distance function to measure the feature discrepancy, such as the MMD [61], a nonparametric metric that measures the distribution divergence between the mean embedding of two distributions in reproducing Kernel Hilbert space (RKHS), and the CORAL [44] computing second-order statistics of two distributions. The DAN [19] and JAN [21] utilize MMD to minimize the distribution in different ways. The DAN minimizes MMD with multiple kernels applied to multiple layers. The JAN aligns the joint distributions of numerous domain-specific layers across domains based on a JMMD criterion. D-CORAL [24] extends CORAL to incorporate it directly into deep networks by constructing a differentiable loss function that minimizes the difference between the source and target correlations. As a case study, we only introduce the two representative approaches (DAN and D-CORAL) to be combined with DLSN to reveal the compatibility and generalization of the proposed DLSN. Other metric discrepancy-based methods are integrated with DLSN in the same way.

The MMD loss is a kernel-based distance function between pairs of samples. The DAN applies the variant of MMD, multiple kernel MMD (MK-MMD) [62], which can leverage different kernels to enhance the MK-MMD test, leading to a principled method for optimal kernel selection. The discrepancy is computed concerning a particular representation ϕ , which is a function mapping the latent feature representation to RKHS $\mathcal{H}$. When the DLSN is integrated with the DAN, an empirical approximation to the discrepancy is computed as follows:

$$l_{\text{mmd}}^2(\mathbf{x}^s, \mathbf{x}^t) = \sum_{v=1}^V \|E_{P_s}[\phi(\mathbf{H}_v^s)] - E_{P_t}[\phi(\mathbf{H}_v^t)]\|_{\mathcal{H}}^2 \quad (8)$$

where E_{P_s} and E_{P_t} are the means of distribution P_s and P_t in RKHS $\mathcal{H}$, respectively. ϕ corresponds to a kernel function

$k(\mathbf{H}_v^s, \mathbf{H}_v^t) := \langle \phi(\mathbf{H}_v^s), \phi(\mathbf{H}_v^t) \rangle_{\mathcal{H}}$, which is defined as the convex combination of m PSD kernels k_u

$$\mathcal{K} = \{k = \sum_{u=1}^m \beta_u k_u : \sum_{u=1}^m \beta_u = 1, \beta_u \geq 0 \quad \forall u\} \quad (9)$$

where the constraints on coefficients β_u are imposed to guarantee that the derived multikernel k is characteristic.

Specifically, CORAL is to align the second-order statistics (covariances) of the source and target features. The D-CORAL is similar to DAN in the sense that a new loss (CORAL loss) is added to minimize the difference in learned feature covariances across domains. The discrepancy loss of implanting the DLSN into D-CORAL is computed as follows:

$$l_{\text{coral}}(\mathbf{x}^s, \mathbf{x}^t) = \frac{1}{4d^2} \|\text{cov}(\mathbf{H}_V^s) - \text{cov}(\mathbf{H}_V^t)\|_F^2 \quad (10)$$

where d is the feature dimension of $\mathbf{H}_s$ and $\mathbf{H}_t$. $\|\cdot\|_F^2$ denotes the squared matrix Frobenius norm. The covariance matrices of the source and target-domain features are given by

$$\text{cov}(\mathbf{H}_V^s) = \frac{1}{n_s - 1} \left((\mathbf{H}_V^s)^T \mathbf{H}_V^s - \frac{1}{n_s} (\mathbf{1}^T \mathbf{H}_V^s)^T (\mathbf{1}^T \mathbf{H}_V^s) \right) \quad (11)$$

$$\text{cov}(\mathbf{H}_V^t) = \frac{1}{n_t - 1} \left((\mathbf{H}_V^t)^T \mathbf{H}_V^t - \frac{1}{n_t} (\mathbf{1}^T \mathbf{H}_V^t)^T (\mathbf{1}^T \mathbf{H}_V^t) \right) \quad (12)$$

where $\mathbf{1}$ is a column vector with all elements equal to 1. n_s and n_t are the samples number of $\mathbf{H}_V^s$ and $\mathbf{H}_V^t$, respectively.

2) *Integrated With Adversarial Learning-Based Methods:* As a case study, we introduce the combination of the DANN [27], [29] and the proposed DLSN, and the combination of the CDAN [33] with the proposed DLSN, respectively. The domain adversarial similarity loss of DANN is utilized for training a model to generate representations such that a domain classifier cannot reliably discriminate the domain of the encoded representation. Maximizing such “indiscrimination” is achieved via a gradient reversal layer (GRL) [27] to predict the domain label producing the hidden representation. The GRL [27] has the same output as the identity function, but reverses the gradient direction. Base on the DANN, the CDAN applies the joint representations of feature and class to deceive the domain classifier. In this way, domain variances in both feature representation and classifier prediction can be

modeled simultaneously. We define the domain classifier as G^{dcls} with the parameter θ^{dcls} . The procedure of integrating the DLSN into DANN or CDAN is as follows:

$$\tilde{\omega} = G^{dcls}(\mathbf{F}^{s/t}) \quad (13)$$

where $\tilde{\omega} \in 0, 1$ is a predictive-domain label of the input sample $\mathbf{x}^s$ or $\mathbf{x}^t$. When the proposed DLSN is integrated with the DANN, $\mathbf{F}^{s/t}$ denotes the representation of source or target domains in the encoder's final layer. When the DLSN combines with CDAN, $\mathbf{F}^{s/t}$ denotes the joint feature representations of the encoder's final layer and the corresponding task classifier. The adversarial loss is defined as follows:

$$l_{adv}(\mathbf{x}^s, \mathbf{x}^t) = \omega \log \tilde{\omega} + (1 - \omega) \log(1 - \tilde{\omega}) \quad (14)$$

where $\omega \in 0, 1$ is the ground truth of the domain label for the input sample.

D. Network Training

The training phase is performed by utilizing labeled source data to learn discriminative transferable features with the classifier and unlabeled target data to reconstruct via the ladder autoencoder, enabling the shared encoder to better focus on learning the cross-domain shared representation. Simultaneously, the distribution discrepancy of the shared features for target and source domains is reduced employing metric embedding or adversarial learning. We optimize the entire network integrated with DLSN end to end, which is described as follows.

The supervised classification of the source domain, the unsupervised reconstruction of the target domain, and the adaption of cross-domain shared features of the two domains are conducted simultaneously. When the DLSN is integrated into the metric discrepancy-based methods, our goal is to minimize the following objective:

$$L_{cls}(\theta^{en}, \theta^{cls}) + \lambda_1 L_{ada}(\theta^{en}, \theta^{cls}) + \lambda_2 L_{rec}(\theta^{en}, \theta^{de}) \quad (15)$$

where λ_1 and λ_2 are the weights that control the interaction of the losses. $L_{cls}(\theta^{en}, \theta^{cls})$ represents a typical cross-entropy loss for classification

$$L_{cls}(\theta^{en}, \theta^{cls}) = \sum_{i=1}^{n_b} E_{\mathbf{x}_i^s, \mathbf{y}_i^s} - \sum_{c=1}^C \mathbb{1}[\mathbf{y}_i^s = c] \log(\tilde{\mathbf{y}}_i^s) \quad (16)$$

where $\mathbb{1}$ is the indicator function. C is the number of classes. $L_{ada}(\theta^{en}, \theta^{cls})$ denotes the adaptation loss of source and target domains

$$L_{ada}(\theta^{en}, \theta^{cls}) = \sum_{i=1}^{n_b} l_{coral}(\mathbf{x}_i^s, \mathbf{x}_i^t) \text{ or } \sum_{i=1}^{n_b} l_{mmd}^2(\mathbf{x}_i^s, \mathbf{x}_i^t). \quad (17)$$

When the DLSN is integrated into the adversarial learning-based methods, our goal is to minimize the following objective:

$$L_{cls}(\theta^{en}, \theta^{cls}) + \lambda_1 L_{ada}(\theta^{en}, \theta^{cls}, \theta^{dcls}) + \lambda_2 L_{rec}(\theta^{en}, \theta^{de}) \quad (18)$$

where $L_{ada}(\theta^{en}, \theta^{cls}, \theta^{dcls})$ denotes the adaptation loss of source and target domains

$$L_{ada}(\theta^{en}, \theta^{cls}, \theta^{dcls}) = \sum_{i=1}^{n_b} l_{adv}(\mathbf{x}_i^s, \mathbf{x}_i^t). \quad (19)$$

The target-domain images are reconstructed using the ladder autoencoder. We define the total reconstruction loss $L_{rec}(\theta^{en}, \theta^{de})$ as the sum of the perceptual loss and pixel reconstruction loss [63], [64] to generate visually indistinguishable images with input images

$$L_{rec}(\theta^{en}, \theta^{de}) = \sum_{i=1}^{n_b} \left(\frac{1}{M_p} \|\hat{\mathbf{x}}_i^t - \mathbf{x}_i^t\|_2^2 + \sum_{l=2}^{L-1} \frac{1}{C_l} \|\mathbf{F}_{il}^t - \hat{\mathbf{F}}_{il}^t\| \right) \quad (20)$$

where n_b is the batch size of target samples. $\mathbf{F}_{il}^t$ is the feature of the target sample $\mathbf{x}_i^t$ at the l th layer of the shared encoder. $\|\cdot\|_2^2$ is the squared L_2 norm. $(1/C_l)$ is the weight to balance perceptual loss and pixel reconstruction loss. C_l denotes the number of channels at the l th layer. M_p is the number of pixels of the input image.

As mentioned above, the supervised classification loss, unsupervised reconstruction loss, and the adaptation loss of domain-shared features are optimized to train the model integrated with DLSN. In this way, we can effectively learn domain-invariant cross-domain shared features to classify by suppressing the domain-specific variations finally.

E. Discussion

The proposed DLSN may be similar to U-Net [65]. Since it was proposed, it has been widely used in location-sensitive tasks, such as segmentation tasks such as semantic/instance/biomedical image segmentation. The essence of U-Net in these tasks is integrating both the local visual cues and global context information existing in multilayers to make a more accurate inference. Intuitively, part of our proposed DLSN architecture, sharing similarity with U-net, there are still inherent differences. First, our target problem is UDA. Our inspiration for adopting this ladder architecture comes from its impressive ability demonstrated in [36]–[38] for semisupervised learning. Second, the way of combining encoder and decoder is different. In U-Net, the features from the encoder and decoder are combined by simply adding them in an elementwise fashion. However, we borrow the idea from [38] to apply a nonlinear combination. The combinator is trainable to adjust the contribution of the encoder and decoder for the reconstruction.

Although the skip connection itself is commonly used in designing network architectures, the insight and motivation underlying it are very different depending on the problem understudied. In the problem of UDA, the proposed DLSN framework is a multitask setting up, and we care about the classification performance on the unlabeled target domain. This ladder-style architecture is introduced to learn the cross-domain shared content by suppressing domain-specific variations. The function of the skip connections in UDA is different from that in segmentation tasks. Here, in



Fig. 5. Example Images from the Digit, Office31, Office-Home, and VisDA-C datasets used in our experiments.

UDA, via the skip connections, the domain-specific components are directly fed to the decoder for reconstruction, relieving the pressure to learn domain-specific variations at the later layers of the shared encoder. Therefore, our DLSN allows the encoder to focus on learning cross-domain shared representations and ignore the domain-specific variations. Nevertheless, according to our knowledge, we are the first to explore the ladder-style architecture for the problem of UDA.

To clarify the contribution of our proposed DLSN, we will discuss three obvious differences between the proposed DLSN and DRCN. First, the motivation is completely different. DRCN aims at learning the domain-invariant features iteratively conducting the supervised classification of the source domain and the unsupervised reconstruction of the target domain. However, there is a problem that DRCN applies all features of an image to adapt. The unsupervised reconstruction of the target domain tries to represent as much information about the original data as possible, whereas supervised learning of the source domain tries to filter out all the information irrelevant for the task at hand. The incompatibility between unsupervised learning and supervised learning [36] in the current network influences the adaptation performance. Compared to DRCN, DLSN applies the ladder layer [36], [39], [40] to suppress the uninformative features for DA. The uninformative features that are needed for the pixel-level image reconstruction task may impair the adaptation performance. The introduced ladder network for UDA can mine the information of the target domain and suppress the domain-specific variations. In this way, we can obtain the cross-domain shared representation. The second difference is the operating mechanism as presented in Section III-B part. We combine the features of the encoder and the decoder layer by layer to relieve the pressure of representing details in the higher layers of the model. The nonlinear combination is a connection, which plays an important role in enabling the proposed DLSN to focus on learning cross-domain shared features while suppressing domain-specific variations. Third, the learning process is also different, as illustrated in Section III-D part. We define the total reconstruction loss as the sum of the perceptual loss and pixel reconstruction loss [36], [63], [64] to generate visually indistinguishable images with input images, which is different from DRCN. By this means, we can ensure that the features of each layer in the encoder are selectively transmitted to the decoder and that the encoder can obtain transferable high-level features layer by layer.

IV. EXPERIMENTS

A. Datasets

To evaluate the effectiveness of the proposed DLSN approach, we conducted extensive experiments on four datasets: 1) Digit; 2) Office31 [66]; 3) Office-Home [67]; and 4) VisDA-C [68]. Fig. 5 shows examples of these four datasets.

Digit includes the MNIST [69], USPS [70], and street view house numbers (SVHNs) [71]. Each dataset contains images of ten classes. MNIST (MN) and USPS (U.S.) are grayscale handwritten digits captured under constrained conditions. The MN dataset contains 60 000 training and 10 000 testing images of size 28×28 , while the U.S. is split into 7291 training and 2007 test images of size 16×16 . SVHN (SV) is a real-world dataset from house numbers in Google Street View images, which contains 73 257 training and 26 032 test RGB images of size 32×32 . We evaluate our methods on the four pairs: U.S. $\rightarrow$ MN, MN $\rightarrow$ U.S., SV $\rightarrow$ MN, and MN $\rightarrow$ SV.

Office-31 is a benchmark dataset for evaluating different DA methods for object recognition, which consists of three different domains: 1) Amazon (A); 2) Dslr (D); and 3) Webcam (W), including 4652 images in 31 classes. All the images are collected from three different domains: 1) Amazon (A), which are downloaded from amazon.com; 2) DSLR (D), which are taken by digital SLR camera; and 3) Webcam (W), which are recorded with a simple webcam. The visual-domain shifts are caused by different photographic settings.

Office-Home is a large benchmark dataset with around 15 500 images and contains images of 65 classes. The dataset consists of images of everyday objects organized into four domains: 1) *artistic images* (Ar): paintings, sketches and/or artistic depictions; 2) *clipart* (Cl): clipart images; 3) *product* (Pr): images without background; and 4) *real world* (Rw): regular images captured with a camera. For each domain, the dataset contains images of 65 object categories, and we can obtain 12 DA tasks.

VisDA-C is a very challenging dataset with the domain shift from synthetic data to real imagery. In this experiment, we validated our method on its classification task. It has two domains, where the Synthetic one consists of 152,397 synthetic 2-D renderings of 3-D objects, and the Real one consists of 55,388 real images from real-world images from MS-COCO [72] dataset. The two domains have 12 classes in common.

B. Network Setting

For the Digit dataset, the backbone architecture contains two pipelines for label prediction and reconstruction, respectively. The label prediction pipeline has five convolutional layers: 1) 100 5×5 filters (Conv1); 2) 150 3×3 filters (Conv2); 3) 150 3×3 filters (Conv3); 4) 150 3×3 filters (Conv4); and 5) 200 3×3 filters (Conv5), and three fully-connected layers (Fc6, Fc7, and Fc8), where Fc8 is the prediction output layer. The number of neurons in Fc6 and Fc7 is 1000 and 500, respectively. There is a ReLU layer behind each hidden layer and output layer. Two max-pooling layers of size 2×2 follow the first and the third ReLU layers (Pool1 and Pool2). The shared encoder has thus a configuration of Conv1 $\rightarrow$ Pool1 $\rightarrow$ Conv2 $\rightarrow$ Conv3 $\rightarrow$ Pool2 $\rightarrow$ Conv4 $\rightarrow$ Conv5. Furthermore, the configuration of the decoder is the inverse of that of the encoder. Note that the downsampling operation in the encoder is replaced by upsampling with nearest-neighbor interpolation. The reconstruction pipeline consists of the shared encoder and the decoder.

For the Office-31, Office-Home, and VisDA-C datasets, we used a standard DCNN trained on the source domain, e.g., ResNet-50 [73] that has been pretrained on ImageNet as the encoder branch of the DLSN. The final classification branch contains a linear layer after global average pooling. For the reconstruction pipeline, we chose the output of the first convolutional layer of ResNet-50 to reconstruct and applied four 3×3 convolutional layers with feature dimensions 1024, 512, 256, and 64 in the decoder of the DLSN. Each convolutional layer is followed by a ReLU nonlinearity and an upsampling layer except the last layer. To prevent overfitting, the dropout layer is added after the first two upsampling layers.

C. Evaluated Methods

For the Digit datasets, we first applied the backbone trained on the source domain to infer the sample of target domain bringing the low bound of the target domain, that is, the performance of Source-only. Similarly, on the Office-31, Office-Home, and VisDA-C datasets, we first used the ResNet-50 [73] to predict samples on the target domain.

Our proposed DLSN can be used as a common module and be integrated with various existing UDA approaches, that is, X + DLSN, to obtain improved performance. Here, we considered state-of-the-art UDA methods, including D-CORAL [24], DAN [19], JAN [21], DANN [29], and CDAN [33], leading to new approaches methods D-CORAL + DLSN, DAN + DLSN, JAN + DLSN, DANN + DLSN, and CDAN + DLSN, respectively.

Besides, to verify the importance of the ladder connections in the autoencoder and demonstrate they can suppress the domain-specific variations, we conducted an ablation study. In other words, we conducted corresponding experiments without ladder connections, that is, D-CORAL + DLSN(w/o L), DAN + DLSN(w/o L), JAN + DLSN(w/o L), DANN + DLSN(w/o L), and CDAN + DLSN(w/o L), where (w/o L) means “without Ladder.” D-CORAL [24]: D-CORAL learns a nonlinear transformation that aligns the second-order statistics of the source

and target distributions in DNNs. DAN [19]: Representations of all task-specific layers in the DAN network are integrated into an RKHS. An optimal multikernel selection method is used to match the mean embedding of two domain distributions to enhance the feature transferability. JAN [21]: The method learns the transformation network by aligning the joint distributions of multiple domain-specific layers across domains based on a JMMD criterion. DANN [29]: The DANN proposes a GRL to train a feature extractor that produces features that maximize the domain binary classifier loss while minimizing the label predictor loss. CDAN [33]: Based on the DANN, the CDAN captures the cross-covariance between feature representations and classifier predictions to improve the discriminability and exploits the discriminative information conveyed in the classifier predictions to assist adversarial adaptation.

D. Implementation Details

We implemented all experiments using Pytorch [74]. For Digit datasets, the network was trained using the ADAM [75] optimizer with a learning rate of 0.0001 and the batch size is 128.

For the Office-31, Office-Home, and VisDA-C datasets, the network was trained using the minibatch stochastic gradient descent (SGD) optimizer with the momentum 0.9. We used a batch size of 28 samples (due to GPU memory constraints). The learning rate annealing strategy was adopted as [29]: $\eta_p = \eta_0(1 + \alpha p)^{-\beta}$, where p was the training progress changing from 0 to 1, $\alpha = 10$, and $\beta = 0.75$ for Office31 and Office-Home, while $\alpha = 10$ and $\beta = 2.25$ for VisDA-C. The η_0 was the initial learning rate that is, 0.0001 for the convolutional layers and 0.001 for the task-specific FC layer.

In particular, for the DAN integrated with DLSN, we used a linear combination of $m = 5$ Gaussian kernels for MK-MMD. For the DANN integrated with DLSN, we added a GRL [27] and one domain classifier with two hidden layers of 1000 and 500 nodes for Digit datasets. For the other three datasets, we added one domain classifier with one hidden layer of 1000 nodes. The coefficient λ_1 was tuned differently to achieve the best results for each approach. We first chose the hyperparameter λ_1 for the D-CORAL, DAN, JAN, DANN, and CDAN following the selection mechanism of these mentioned algorithms. Then, to be fair, the selected hyperparameter λ_1 was applied when these algorithms were integrated with the proposed DLSN or DLSN(w/o L). About the Digit dataset, the coefficient λ_1 was set to 0.3 for D-CORAL, DAN, D-CORAL + DLSN(w/o L), DAN + DLSN(w/o L), D-CORAL + DLSN, and DAN + DLSN, the coefficient of which was set to 1 on Office-31, Office-Home, and VisDA-C datasets. For JAN, DANN, JAN + DLSN(w/o L), DANN + DLSN(w/o L), JAN + DLSN, and DANN + DLSN, the hyperparameter λ_1 was changed from 0 to 1 by a progressive schedule $\lambda_p = [2/(1 + \exp(-\gamma p))] - 1$, and $\gamma = 10$ on all datasets [29]. For CDAN, CDAN + DLSN(w/o L), and CDAN + DLSN, λ_1 was set to 1 for all datasets. We conduct importance-weighted cross-validation (IWCV) [76] following CDAN to select λ_2 . As D-CORAL + DLSN, DAN + DLSN, JAN + DLSN, DANN + DLSN, and CDAN + DLSN perform

TABLE I

CLASSIFICATION ACCURACIES (%) OF VARIOUS METHODS ON DIGIT DATASETS FOR UDA. BOLD NUMBERS AND UNDERLINED NUMBERS DENOTE THE BEST RESULTS AND THE SECOND-BEST RESULTS FOR EACH COLUMN, RESPECTIVELY. THE NUMBER IN () DENOTES THE RISING VALUE COMPARED TO THE CORRESPONDING BASELINE METHODS

	MN→US	US→MN	SV→MN	Average
Source only	86.3	87.9	73.5	82.6
Compared Methods				
ADDA [28]	89.4	90.1	76.0	85.2
MCD [48]	96.5	94.1	<u>96.2</u>	95.6
CoGAN [51]	91.2	89.1	No Conv.	—
PixelDA [50]	95.9	—	—	—
DTN [52]	—	—	84.4	—
UNIT [55]	95.9	93.5	90.5	93.3
GTA [53]	92.5	90.8	84.7	89.3
SBADA-GAN [57]	97.6	95.0	76.1	89.6
CyCADA [54]	94.8	95.7	88.3	92.9
I2I [58]	95.1	92.2	92.1	93.1
DRCN [35]	91.8	73.7	82.0	82.5
DSN [59]	91.3	—	82.7	—
BSP+CDAN [78]	95.0	98.1	92.1	95.1
DWT-MEC [77]	99.3	99.2	99.1	99.2
Evaluated Methods				
D-CORAL [24]	94.9	95.5	89.9	93.4
D-CORAL+DLSN(w/o L)	95.4	96.2	91.4	94.3
D-CORAL+DLSN	97.6 (↑2.7)	96.6 (↑1.1)	94.7 (↑4.8)	96.3 (↑2.9)
DAN [19]	87.2	96.2	84.1	89.2
DAN+DLSN(w/o L)	92.9	96.3	89.1	92.8
DAN+DLSN	94.5 (↑7.3)	96.5 (↑0.3)	89.7 (↑5.6)	93.6 (↑4.4)
JAN [21]	90.1	94.2	84.3	89.5
JAN+DLSN(w/o L)	91.7	94.6	87.6	91.3
JAN+DLSN	93.1 (↑3.0)	95.3 (↑1.1)	88.8 (↑4.5)	92.4 (↑2.9)
DANN [29]	85.9	90.9	74.2	83.7
DANN+DLSN(w/o L)	87.0	92.4	No Conv.	—
DANN+DLSN	89.6 (↑3.7)	92.7 (↑1.8)	78.1 (↑3.9)	86.8 (↑3.1)
CDAN [33]	93.9	97.2	88.5	93.2
CDAN+DLSN(w/o L)	95.1	98.0	90.1	94.4
CDAN+DLSN	95.8 (↑1.7)	<u>98.3</u> (↑1.1)	91.2 (↑2.7)	95.1 (↑1.9)

stably under several different parameters [0.1, 0.5, 1], we directly fix λ_2 to 1.

E. Results

In this section, we conducted extensive experiments to demonstrate the effectiveness of the proposed DLSN. To be fair, for Digit datasets, we implemented experiments for all methods because the performance of our architecture outperforms the report results of the previous papers. For Office31 and Office-Home datasets, we cited the experimental results from the original papers. For VisDA-C, we regarded ResNet-50 as the base feature extractor. We implemented evaluated methods: D-CORAL,¹ DAN, JAN,² CDAN,³ and DANN⁴ and compared methods: MCD⁵ and BSP+CDAN⁶ according to the released code.

Results on the Digit Dataset: The classification accuracy on the Digit datasets for UDA is shown in Table I. We can see the proposed method, e.g., D-CORAL + DLSN, outperforms most algorithms. DWT-MEC [77] obtains the best performance due to multiple domain-specific alignment layers. Notably, on the MN→U.S. and U.S.→MN tasks, D-CORAL + DLSN

TABLE II

CLASSIFICATION ACCURACIES (%) ON THE OFFICE31 DATASET FOR UDA. ALL MODELS UTILIZE RESNET-50 AS THE BASE ARCHITECTURE. THE BOLD NUMBERS DENOTE THE BEST RESULTS FOR EACH COLUMN. THE NUMBER IN () DENOTES THE RISING VALUE COMPARED TO THE CORRESPONDING BASELINE METHODS

	A→W	D→W	W→D	A→D	D→A	W→A	Average
ResNet-50 [73]	68.4	96.7	99.3	68.9	62.5	60.7	76.1
Compared Methods							
MADA [32]	90.0	97.4	99.6	87.8	70.3	66.4	85.2
SimNet [79]	88.6	98.2	99.7	85.3	73.4	71.6	86.2
GTA [53]	89.5	97.9	99.8	87.7	72.8	71.4	86.5
MCD [48]	88.6	98.5	100.0	92.2	69.5	69.7	86.5
MSTN [45]	91.3	98.9	100.0	90.4	72.7	65.6	86.5
MSTN+DSBN [80]	92.7	99.0	100.0	92.2	71.7	74.4	88.3
DAAA [56]	86.8	99.3	100.0	88.8	74.3	73.9	87.2
iCAN [81]	92.5	98.8	100.0	90.1	72.1	69.9	87.2
TAT [46]	92.5	99.3	100.0	93.2	73.1	72.1	88.4
BSP+CDAN [78]	93.3	98.2	100.0	93.0	73.6	72.6	88.5
Evaluated Methods							
D-CORAL [24]	79.5	97.4	99.8	80.9	63.5	62.7	80.6
D-CORAL+DLSN(w/o L)	80.6	97.8	100.0	82.1	63.1	63.0	81.1
D-CORAL+DLSN	81.8 (↑2.3)	98.7 (↑1.3)	100.0 (↑0.2)	83.3 (↑2.4)	64.2 (↑0.7)	64.1 (↑1.4)	82.0 (↑1.4)
DAN [19]	80.5	97.1	99.6	78.6	63.6	62.8	80.4
DAN+DLSN(w/o L)	84.3	97.7	100.0	85.3	64.1	64.5	82.7
DAN+DLSN	85.5 (↑5.0)	98.2 (↑1.1)	100.0 (↑0.4)	86.4 (↑7.8)	65.3 (↑1.7)	65.5 (↑2.7)	83.5 (↑3.1)
JAN [21]	85.4	97.4	99.8	84.7	68.6	70.0	84.3
JAN+DLSN(w/o L)	88.8	97.9	99.8	88.5	67.2	68.9	85.2
JAN+DLSN	90.9 (↑5.5)	98.2 (↑0.8)	100.0 (↑0.2)	89.6 (↑4.9)	70.3 (↑1.7)	71.2 (↑1.2)	86.7 (↑2.4)
DANN [29]	82.0	96.9	99.1	79.7	68.2	67.4	82.2
DANN+DLSN(w/o L)	82.6	97.1	99.7	83.5	70.2	68.6	83.7
DANN+DLSN	84.8 (↑2.8)	97.8 (↑0.9)	100.0 (↑0.9)	86.5 (↑6.8)	72.3 (↑4.1)	70.6 (↑3.2)	85.3 (↑3.1)
CDAN [33]	93.1	98.2	100.0	89.8	70.1	68.0	86.6
CDAN+DLSN(w/o L)	93.6	98.7	100.0	90.1	72.0	72.8	87.9
CDAN+DLSN	94.8 (↑1.7)	99.4 (↑1.2)	100.0 (↑0.0)	93.3 (↑3.5)	72.8 (↑2.7)	73.3 (↑5.3)	88.9 (↑2.3)

and CDAN + DLSN achieve the second-best performance, respectively. Even so, the experimental methods integrated with DLSN outperform the state-of-the-art approaches on the whole.

To further analyze, the performance of many baseline methods obtains improvement via combining the proposed DLSN, which demonstrates the efficiency and generalization of the DLSN. It should be noted that on the SV→MN task, DANN + DLSN(w/o L) cannot converge. However, when the ladder connection is adopted, the model gets convergence, and the performance obtains the improvement.

Results on Office31: Table II shows the detailed comparison results of these approaches in 6 transfer tasks. From Table II, we can see that in almost all domain-adaptation tasks, all baseline methods integrated with the DLSN produce better recognition accuracy. Especially, for these two tasks A→W and A→D, the improvement of integrated methods over baseline methods is by a large margin. The adaptation from simple images to complex images is difficult. The images from the background of the Amazon (A) domain are relatively clear, while the images taken in the real world from Webcam (W) and Dslr (D) domains are with a certain degree of complexity. When transferring from the Amazon (A) domain to Webcam (W) or Dslr (D) domains, the proposed method reconstructs the target domain to learn the potential characteristics of the target domain and further uses the ladder connection to suppress the redundant information of the complex target domain. In this way, the network architecture not only mines the characteristics of the target domain but also suppresses domain-specific information. From the results of these two tasks A→W and A→D, we can see the proposed method has a great advantage of dealing with the adaptation from a simple domain to a complex domain.

Besides, from Table II, we can find experimental methods integrated with DLSN(w/o L) achieve a specific improvement in some cases, which proves that mining the information of the target domain is indeed valuable. However, in some tasks, such as W→A and D→A, the performance of some experimental

¹<https://github.com/VisionLearningGroup/CORAL>

²<https://github.com/thuml/Xlearn>

³<https://github.com/thuml/CDAN>

⁴<http://sites.skoltech.ru/compvision/projects/grl/>

⁵https://github.com/mil-tokyo/MCD_DA

⁶<https://github.com/thuml/Batch-Spectral-Penalization>

TABLE III
CLASSIFICATION ACCURACIES (%) ON THE OFFICE-HOME DATASET FOR UDA. ALL MODELS UTILIZE RESNET-50 AS BASE ARCHITECTURE. THE BOLD NUMBERS DENOTE THE BEST RESULTS FOR EACH COLUMN. THE NUMBERS IN () DENOTE THE RISING VALUE COMPARED TO THE CORRESPONDING BASELINE METHODS

	Ar→Cl	Ar→Pr	Ar→Rw	Cl→Ar	Cl→Pr	Cl→Rw	Pr→Ar	Pr→Cl	Pr→Rw	Rw→Ar	Rw→Cl	Rw→Pr	Average
ResNet-50 [73]	34.9	50.0	58.0	37.4	41.9	46.2	38.5	31.2	60.4	53.9	41.2	59.9	46.1
Compared Methods													
SE [82]	48.8	61.8	72.8	54.1	63.2	65.1	50.6	49.2	72.3	66.1	55.9	78.7	61.5
DWT-MEC [77]	50.3	72.1	77.0	59.6	69.3	70.2	58.3	48.1	77.3	69.3	53.6	82.0	65.6
TAT [46]	51.6	69.5	75.4	59.4	69.5	68.6	59.5	50.5	76.8	70.9	56.6	81.6	65.8
BSP+CDAN [78]	52.0	68.6	76.1	58.0	70.3	70.2	58.6	50.2	77.6	72.2	59.3	81.9	66.3
SAFN [83]	52.0	71.7	76.3	64.2	69.9	71.9	63.7	51.4	77.1	70.9	57.1	81.5	67.3
Evaluated Methods													
D-CORAL [24]	43.8	64.8	73.0	55.8	63.1	64.6	40.7	39.6	68.7	64.6	43.8	75.7	58.1
D-CORAL+DLSN(w/o L)	46.9	67.1	75.8	57.5	66.0	67.5	52.4	42.1	72.9	67.7	47.9	77.8	61.8
D-CORAL+DLSN	47.6(↑3.2)	67.4(↑2.6)	76.8(↑3.8)	58.6(↑2.8)	67.4(↑4.3)	68.8(↑4.2)	54.0(↑13.3)	42.8(↑3.2)	74.0(↑5.3)	68.5(↑3.9)	49.4(↑5.6)	78.9(↑3.2)	62.9(↑4.8)
DAN [19]	43.6	57.0	67.9	45.8	56.5	60.4	44.0	43.6	67.7	63.1	51.5	74.3	56.3
DAN+DLSN(w/o L)	50.3	66.7	73.8	54.5	62.0	64.2	53.1	45.8	74.0	63.6	50.9	77.2	61.3
DAN+DLSN	52.6(↑9.0)	68.1(↑11.1)	75.7(↑7.8)	60.6(↑14.8)	66.9(↑10.4)	69.6(↑9.2)	58.5(↑14.5)	51.1(↑7.5)	76.1(↑8.4)	64.9(↑1.8)	52.7(↑1.2)	78.4(↑4.1)	64.6(↑8.3)
JAN [21]	45.9	61.2	68.9	50.4	59.7	61.0	45.8	43.4	70.3	63.9	52.4	76.8	58.3
JAN+DLSN(w/o L)	48.8	67.9	74.2	60.0	66.2	66.7	58.6	47.4	76.8	69.2	53.1	80.6	64.1
JAN+DLSN	49.8(↑3.9)	69.5(↑8.3)	75.8(↑6.9)	60.8(↑10.4)	67.4(↑7.7)	68.3(↑7.3)	60.4(↑14.6)	48.8(↑5.4)	78.9(↑8.6)	70.4(↑6.5)	54.5(↑2.1)	81.3(↑4.5)	65.5(↑7.2)
DANN [29]	45.6	59.3	70.1	47.0	58.5	60.9	46.1	43.7	68.5	63.2	51.8	76.8	57.6
DANN+DLSN(w/o L)	44.3	63.9	72.3	52.9	61.5	63.5	51.4	42.8	72.5	65.9	50.7	77.2	59.9
DANN+DLSN	46.6(↑1.0)	65.8(↑6.5)	73.5(↑3.4)	54.4(↑7.4)	62.3(↑3.8)	65.6(↑4.7)	52.5(↑6.4)	44.2(↑0.5)	73.1(↑4.6)	66.3(↑3.1)	51.9(↑0.1)	77.7(↑0.9)	61.2(↑3.6)
CDAN [33]	49.0	69.3	74.5	54.4	66.0	68.4	55.6	48.3	75.9	68.4	55.4	80.5	63.8
CDAN+DLSN(w/o L)	50.6	69.4	75.1	57.4	68.6	68.9	60.7	52.0	77.9	72.6	58.0	81.8	66.1
CDAN+DLSN	52.7(↑3.7)	70.1(↑0.8)	77.2(↑2.7)	59.8(↑5.4)	69.3(↑3.3)	70.2(↑1.8)	61.6(↑6.0)	54.0(↑5.7)	78.9(↑3.0)	72.9(↑4.5)	59.3(↑3.9)	82.2(↑1.7)	67.4(↑3.6)

TABLE IV
CLASSIFICATION ACCURACY (%) OF EACH CATEGORY ON THE VISDA-C DATASET FOR UDA. ALL MODELS UTILIZE RESNET-50 AS THE BASE ARCHITECTURE. THE BOLD NUMBERS DENOTE THE BEST RESULTS FOR EACH COLUMN. THE NUMBER IN () DENOTES THE RISING VALUE COMPARED TO THE CORRESPONDING BASELINE METHODS

	airplane	bicycle	bus	car	horse	knife	motorcycle	person	plant	skateboard	train	truck	Average
ResNet-50 [73]	83.1	29.4	63.4	64.7	70.8	5.5	87.4	28.1	76.1	40.3	87.4	20.1	54.7
Compared Methods													
MCD [48]	83.6	66.9	86.3	70.5	85.6	26.7	85.1	67.1	72.9	73.3	82.2	12.6	67.7
BSP+CDAN [78]	90.1	66.5	76.8	49.6	85.9	22.2	86.2	73.3	83.5	76.8	82.0	31.9	68.7
Evaluated Methods													
D-CORAL [24]	88.3	19.2	72.3	62.7	66.7	4.9	92.6	51.4	80.4	31.4	88.9	24.2	56.9
D-CORAL+DLSN(w/o L)	88.3	19.4	73.8	63.9	66.9	5.8	92.4	54.1	79.2	36.2	89.7	25.8	57.9
D-CORAL+DLSN	89.6(↑1.3)	20.6(↑1.4)	75.0(↑2.7)	64.9(↑2.2)	68.2(↑1.5)	6.0(↑1.1)	94.6(↑2.0)	60.5(↑9.1)	81.9(↑1.5)	36.8(↑5.4)	90.1(↑1.2)	26.6(↑2.4)	59.6(↑2.7)
DAN [19]	81.6	24.8	65.8	65.8	74.6	5.6	93.4	57.5	80.9	46.7	86.3	17.6	58.4
DAN+DLSN(w/o L)	82.5	25.5	66.7	68.3	75.3	6.5	94.1	58.1	81.3	47.4	88.1	17.9	59.3
DAN+DLSN	89.8(↑8.2)	34.1(↑9.3)	66.3(↑0.5)	67.1(↑1.3)	77.2(↑2.6)	11.9(↑6.3)	94.5(↑1.1)	60.6(↑3.1)	82.0(↑1.1)	47.9(↑1.2)	88.6(↑2.3)	18.2(↑0.6)	61.5(↑3.1)
DANN [29]	85.2	33.8	60.4	70.1	72.3	4.6	90.8	28.9	68.6	48.3	87.1	11.4	55.3
DANN+DLSN(w/o L)	85.4	43.7	65.7	65.6	74.0	5.8	90.7	28.4	70.3	47.6	88.2	14.6	56.7
DANN+DLSN	86.5(↑1.3)	46.3(↑2.5)	71.3(↑10.9)	68.2(↑1.9)	75.9(↑3.6)	6.3(↑1.7)	91.2(↑0.4)	30.7(↑1.8)	71.4(↑2.8)	49.2(↑0.9)	88.5(↑1.4)	15.4(↑4.0)	58.4(↑3.1)
JAN [21]	86.9	47.7	66.7	65.2	80.1	16.6	84.8	63.6	87.5	37.0	87.8	15.6	61.6
JAN+DLSN(w/o L)	89.8	48.3	81.5	64.9	81.8	18.9	92.9	67.5	92.3	38.2	88.2	13.4	64.8
JAN+DLSN	90.7(↑3.8)	48.7(↑1.0)	82.0(↑15.3)	70.4(↑5.2)	82.2(↑2.1)	20.2(↑3.6)	93.1(↑8.3)	68.9(↑5.3)	93.6(↑6.1)	47.1(↑10.1)	88.7(↑0.9)	16.4(↑1.2)	66.8(↑5.2)
CDAN [33]	86.1	34.8	61.5	63.7	76.5	27.8	90.3	57.4	84.6	60.9	89.6	34.2	63.9
CDAN+DLSN(w/o L)	84.9	41.4	62.3	65.9	82.6	33.2	93.9	59.9	87.3	66.5	89.2	36.5	67.0
CDAN+DLSN	87.8(↑1.7)	44.5(↑9.7)	67.0(↑5.5)	73.3(↑9.6)	84.7(↑8.2)	34.8(↑7.0)	94.6(↑4.3)	59.2(↑1.8)	86.7(↑2.1)	68.6(↑7.7)	90.1(↑0.5)	44.1(↑9.9)	69.6(↑5.7)

methods integrated with DLSN(w/o L) drops. This may be caused by the compulsive alignment of task-specific features from the source domain and all information from the target domain. With the ladder connections, all experimental methods achieve a performance improvement, verifying the availability and generality of the proposed DLSN. Significantly, the combination of the proposed DLSN and CDAN outperforms the other methods and achieves state-of-the-art performance.

Results on Office-Home: Table III shows the results of our experiments on the Office-Home dataset. The Office-Home dataset is a very challenging dataset for DA due to the large categories in each domain. From the results, we have the following observations.

- 1) The DAN + DLSN(w/o L) and JAN + DLSN(w/o L) outperform DAN and JAN on most transfer tasks by a large margin and even exceed these two baseline methods at least 5%, respectively, which confirms that mining of the target data structure is vital in the DA problem. Besides, the DAN + DLSN and JAN + DLSN further improve the performance, which indicates suppressing

the domain-specific variations via lateral connections is beneficial to the DA problem.

- 2) The CDAN + DLSN performs best and exceeds the combination of the BSP [78] and CDAN by 1.1%.

From Table III, we can see the experimental methods combined with DLSN dramatically improve the performance on most adaptation tasks, demonstrating the efficiency of the proposed DLSN in suppressing the domain-specific variations. In all, our proposed framework, called DLSN, can be applied to current DA algorithms successfully and dramatically improves performance.

Results on VisDA-C: Table IV lists the accuracy over 12 classes on VisDA-C with the validation set as the target domain. Our proposed method improves accuracy consistently by applying DLSN to the baseline models. The mean accuracy of the model integrated by DLSN outperforms the baseline methods by more than 2%. It is worth noting that the experimental methods only with reconstruction achieve improvements in the performance, demonstrating the importance of mining the target domain. It is difficult for all models to recognize hard classes such as a knife, person, and truck.

TABLE V
OVERALL CLASSIFICATION ACCURACIES (%) ON THE VISDA-C DATASET FOR UDA. ALL MODELS UTILIZE RESNET-50
AS THE BASE ARCHITECTURE. THE BOLD NUMBER DENOTES THE BEST RESULT

Methods	ResNet-50 [73]	D-CORAL [24]	DAN [19]	DANN [29]	JAN [21]	CDAN [33]	BSP+CDAN [78]	MCD [48]	CDAN+DLSN
Average	60.0	61.3	63.1	61.7	64.8	66.8	68.2	69.2	70.8

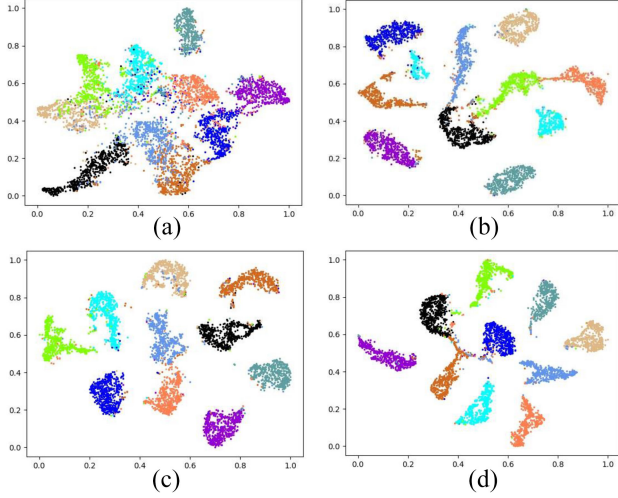


Fig. 6. (a)–(d) t-SNE visualization of network activations on target domain MNIST generated by Source-only, D-CORAL, D-CORAL + DLSN(w/o L), and D-CORAL + DLSN. Different colors represent different classes.

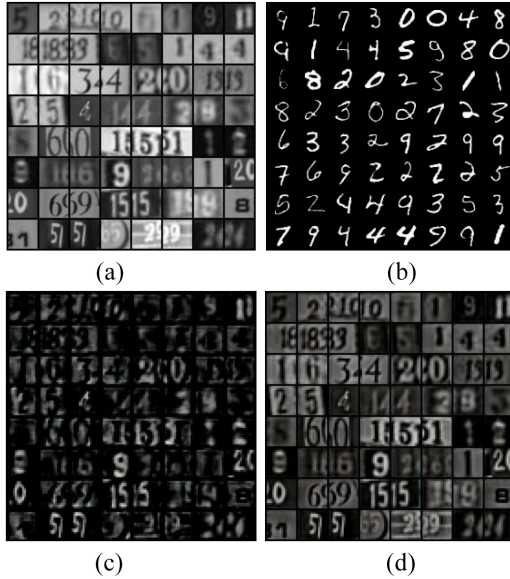


Fig. 7. Data reconstruction after training from SVHN to MNIST. (a) and (b) Samples from source (SVHN) and target (MNIST). (c) Reconstruction of SVHN using D-CORAL + DLSN(w/o L). (d) Reconstruction of SVHN using D-CORAL + DLSN.

However, the methods combined with the proposed DLSN alleviate some difficulties.

To have a complete comparison with most existing methods, Table V presents the overall accuracy on VisDA-C datasets. We can see CDAN + DLSN outperforms BSP+CDAN [78] and MCD [48]. As a whole, the methods integrated with DLSN can achieve state-of-the-art performance when combined with DLSN.

F. Further Remarks

Feature Visualization: A popular method to visualize high-dimensional data in 2-D is t-SNE [84]. We are interested in the distribution of embeddings for the target domain when we employ our training scheme. We chose the complicated SV→MN DA task of the Digit dataset and plotted in Fig. 6 the t-SNE visualization to visualize the learned feature representations. Fig. 6(a) depicts the learned feature of the target domain by Source-only. We can see that the distribution of the target sample is disordered for the Source-only run without DA. Fig. 6(b) shows the visualizations of features learned by D-CORAL. Some same classes can gather together, but the feature distribution is still discrete. The feature representation of D-CORAL + DLSN(w/o L) is described in Fig. 6(c). We can see that the clusters become compact, but some categories have been mixed up in the feature space. As shown in Fig. 6(d), the D-CORAL + DLSN features on the target domain clusters and can be discriminated very well. In contrast, combining DLSN can learn more discriminative representations, which jointly enlarges the interclass dispersion and reduces the intraclass variations.

Reconstruction Analysis: In order to analyze the ladder network further, we reconstructed the source domain with the trained model as shown in Fig. 7. Notably, all trained models only reconstruct the target domain images during the training process. Fig. 7(a) and (b) denotes images of the source domain (SVHN) and the target domain (MNIST), respectively. Fig. 7(c) shows the reconstruction result of the source domain with D-CORAL + DLSN(w/o L). Some digits can be reconstructed, but some details are lost, which verifies the contradiction between unsupervised reconstruction and supervised classification learning at the same time via the shared encoder. Fig. 7(d) is the reconstruction result of D-CORAL + DLSN, completely reconstructing the information of the source domain, which demonstrates the shared encoder learns the domain-invariant features for the source and target domains, and the ladder connections can suppress the domain-specific features.

Convergence: We testified the performance of experimental methods on the convergence from the testing loss, which is illustrated in Fig. 8(a)–(e). Our model has a similar convergence speed as the baseline methods. We can see that combining DLSN leads to a notable loss decrease of the original methods from the loss curve. From the beginning of training, the performance of approaches integrated with DLSN performs better. As the training proceeds, the loss of the methods combining DLSN has always been at a lower level, which implies that DLSN effectively improves the performance.

Discrepancy Distance: The theory of DA [18], [85] denotes the $\mathcal{A}$ -distance as a measure of a cross-domain discrepancy, which will bound the target risk together with the source

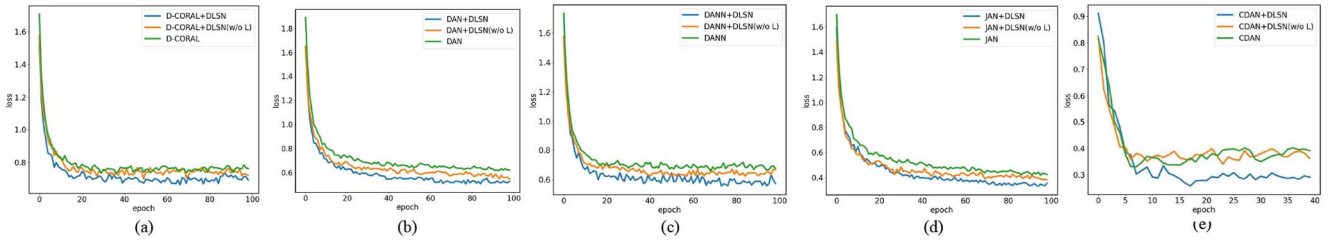


Fig. 8. Empirical analysis: (a)–(e) Curve of the classification loss during training on task A→W of the Office-31 dataset.

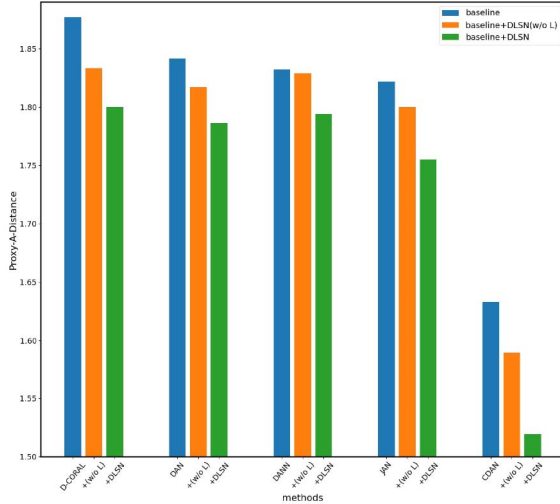


Fig. 9. Empirical analysis: Proxy $\mathcal{A}$ -Distance of different features.

risk. The way of estimating the proxy $\mathcal{A}$ -distance (PAD) is defined as $\hat{d}_{\mathcal{A}} = 2(1 - 2\epsilon)$, where ϵ is the generalization error of a binary classifier of discriminating source and target. We applied a kernel SVM to estimate the $\mathcal{A}$ -distance. Fig. 9 shows PADs on tasks A→W with features of baseline methods, baseline methods + DLSN(w/o L), and baseline methods integrated with the proposed DLSN. We observe that the PAD of experimental approaches integrated with the proposed DLSN is much smaller than the baseline methods, suggesting that our features can reduce the cross-domain gap more effectively.

V. CONCLUSION

In this article, we proposed a new module, called DLSN, for the problem of UDA. It aims at extracting expressive cross-domain shared representations by suppressing domain-specific variations that may hurt target-domain classification performance. The proposed DLSN can be used as a common module to integrate with various existing UDA frameworks to further boost the performance. The extensive experimental evaluation clearly showed the effectiveness of DLSN in significantly improving the performance of popular UDA approaches. Importantly, our study showed that the lateral connections of the reconstruction autoencoder, which connect the shared encoder to the decoder play a vital role in learning better cross-domain shared features.

REFERENCES

- [1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. NIPS*, 2012, pp. 1106–1114.
- [2] O. Russakovsky *et al.*, "ImageNet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, 2015.
- [3] L. Liu *et al.*, "Deep learning for generic object detection: A survey," *Int. J. Comput. Vis.*, vol. 128, no. 2, pp. 261–318, 2020.
- [4] S. Minaee, Y. Boykov, F. Porikli, A. Plaza, N. Kehtarnavaz, and D. Terzopoulos, "Image segmentation using deep learning: A survey," 2020. [Online]. Available: arXiv:1702.05374
- [5] A. García-García, S. Orts-Escobedo, S. Oprea, V. Villena-Martínez, P. Martínez-González, and J. García-Rodríguez, "A survey on deep learning techniques for image and video semantic segmentation," *Appl. Soft Comput.*, vol. 70, pp. 41–65, Sep. 2018.
- [6] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, "DeepFace: Closing the gap to human-level performance in face verification," in *Proc. CVPR*, 2014, pp. 1701–1708.
- [7] T. Tommasi, N. Patricia, B. Caputo, and T. Tuytelaars, "A deeper look at dataset bias," in *Proc. GCPR*, 2015, pp. 504–516.
- [8] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?" in *Proc. NIPS*, 2014, pp. 3320–3328.
- [9] J. Donahue *et al.*, "Decaf: A deep convolutional activation feature for generic visual recognition," in *Proc. ICML*, 2014, pp. 647–655.
- [10] Y. Chen, S. Song, S. Li, L. Yang, and C. Wu, "Domain space transfer extreme learning machine for domain adaptation," *IEEE Trans. Cybern.*, vol. 40, no. 5, pp. 1909–1922, May 2019.
- [11] Z. Cui, W. Li, D. Xu, S. Shan, X. Chen, and X. Li, "Flowing on riemannian manifold: Domain adaptation by shifting covariance," *IEEE Trans. Cybern.*, vol. 44, no. 12, pp. 2264–2273, Dec. 2014.
- [12] S. Khalighi, B. Ribeiro, and U. J. Nunes, "Importance weighted import vector machine for unsupervised domain adaptation," *IEEE Trans. Cybern.*, vol. 47, no. 10, pp. 3280–3292, Oct. 2017.
- [13] M. Wang and W. Deng, "Deep visual domain adaptation: A survey," *Neurocomputing*, vol. 312, pp. 135–153, Oct. 2018.
- [14] G. Csúrká, "Domain adaptation for visual applications: A comprehensive survey," 2017. [Online]. Available: arXiv:1702.05374
- [15] W. M. Kouw and M. Loog, "A review of domain adaptation without target labels," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 3, pp. 766–785, Mar. 2021.
- [16] Z. Lei, "Transfer adaptation learning: A decade survey," 2019. [Online]. Available: arXiv:1903.04687
- [17] S. Ben-David, J. Blitzer, K. Crammer, and F. Pereira, "Analysis of representations for domain adaptation," in *Proc. NIPS*, 2007, pp. 137–144.
- [18] S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. W. Vaughan, "A theory of learning from different domains," *Mach. Learn.*, vol. 79, nos. 1–2, pp. 151–175, 2010.
- [19] M. Long, Y. Cao, J. Wang, and M. I. Jordan, "Learning transferable features with deep adaptation networks," in *Proc. ICML*, 2015, pp. 97–105.
- [20] M. Ghifary, W. B. Kleijn, and M. Zhang, "Domain adaptive neural networks for object recognition," in *Proc. PRICAI*, 2014, pp. 898–904.
- [21] M. Long, H. Zhu, J. Wang, and M. I. Jordan, "Deep transfer learning with joint adaptation networks," in *Proc. 34th ICML*, vol. 70, 2017, pp. 2208–2217.
- [22] W. Zellinger, T. Grubinger, E. Lughofer, T. Natschlager, and S. Saminger-Platz, "Central moment discrepancy (CMD) for domain-invariant representation learning," 2017. [Online]. Available: arXiv:1702.08811
- [23] M. Long, Y. Cao, Z. Cao, J. Wang, and M. I. Jordan, "Transferable representation learning with deep adaptation networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 12, pp. 3071–3085, Dec. 2019.

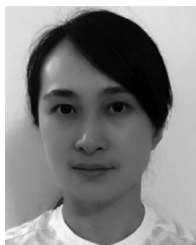
- [24] B. Sun and K. Saenko, "Deep coral: Correlation alignment for deep domain adaptation," in *Proc. ECCV*, 2016, pp. 443–450.
- [25] M. Long, H. Zhu, J. Wang, and M. I. Jordan, "Unsupervised domain adaptation with residual transfer networks," in *Proc. NIPS*, 2016, pp. 136–144.
- [26] H. Yan, Z. Li, Q. Wang, P. Li, Y. Xu, and W. Zuo, "Weighted and class-specific maximum mean discrepancy for unsupervised domain adaptation," *IEEE Trans. Multimedia*, vol. 22, no. 9, pp. 2420–2433, Sep. 2020.
- [27] Y. Ganin and V. S. Lempitsky, "Unsupervised domain adaptation by backpropagation," in *Proc. ICML*, 2015, pp. 1180–1189.
- [28] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell, "Adversarial discriminative domain adaptation," in *Proc. CVPR*, 2017, pp. 7167–7176.
- [29] Y. Ganin *et al.*, "Domain-adversarial training of neural networks," *J. Mach. Learn. Res.*, vol. 17, no. 1, pp. 2096–2030, 2016.
- [30] J. Shen, Y. Qu, W. Zhang, and Y. Yu, "Wasserstein distance guided representation learning for domain adaptation," in *Proc. AAAI*, 2018, pp. 4058–4065.
- [31] E. Tzeng, J. Hoffman, T. Darrell, and K. Saenko, "Simultaneous deep transfer across domains and tasks," in *Proc. ICCV*, 2015, pp. 4068–4076.
- [32] Z. Pei, Z. Cao, M. Long, and J. Wang, "Multi-adversarial domain adaptation," in *Proc. AAAI*, 2018, pp. 3934–3941.
- [33] M. Long, Z. Cao, J. Wang, and M. I. Jordan, "Conditional adversarial domain adaptation," *Proc. NIPS*, 2018, pp. 1647–1657.
- [34] G. Ian *et al.*, "Generative adversarial nets," in *Proc. NIPS*, 2014, pp. 2672–2680.
- [35] M. Ghifary, W. B. Kleijn, M. Zhang, D. Balduzzi, and W. Li, "Deep reconstruction-classification networks for unsupervised domain adaptation," in *Proc. ECCV*, 2016, pp. 597–613.
- [36] H. Valpola, "From neural PCA to deep unsupervised learning," in *Advances in Independent Component Analysis and Learning Machines*. London, U.K.: Academic, 2015, pp. 143–171.
- [37] A. Rasmus, M. Berglund, M. Honkala, H. Valpola, and T. Raiko, "Semi-supervised learning with ladder networks," in *Proc. NIPS*, 2015, pp. 3546–3554.
- [38] M. Pezeshki, L. Fan, P. Brakel, A. Courville, and Y. Bengio, "Deconstructing the ladder network architecture," in *Proc. ICML*, 2016, pp. 2368–2376.
- [39] T. Jeong, Y. Lee, and H. Kim, "Ladder capsule network," in *Proc. ICML*, vol. 97, 2019, pp. 3071–3079.
- [40] I. Prémont-Schwarz, A. Ilin, T. Hao, A. Rasmus, R. Boney, and H. Valpola, "Recurrent ladder networks," in *Proc. NeurIPS*, 2017, pp. 6009–6019.
- [41] E. Tzeng, J. Hoffman, N. Zhang, K. Saenko, and T. Darrell, "Deep domain confusion: Maximizing for domain invariance," 2014. [Online]. Available: arXiv:1412.3474
- [42] Y. Hongliang, D. Yukang, L. Peihua, W. Qilong, X. Yong, and Z. Wangmeng, "Mind the class weight bias: Weighted maximum mean discrepancy for unsupervised domain adaptation," in *Proc. CVPR*, 2017, pp. 2272–2281.
- [43] G. Kang, L. Jiang, Y. Yang, and A. G. Hauptmann, "Contrastive adaptation network for unsupervised domain adaptation," in *Proc. CVPR*, 2019, pp. 4893–4902.
- [44] B. Sun, J. Feng, and K. Saenko, "Return of frustratingly easy domain adaptation," in *Proc. AAAI*, 2016, pp. 2058–2065.
- [45] S. Xie, Z. Zheng, L. Chen, and C. Chen, "Learning semantic representations for unsupervised domain adaptation," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 5423–5432.
- [46] H. Liu, M. Long, J. Wang, and M. Jordan, "Transferable adversarial training: A general approach to adapting deep classifiers," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 4013–4022.
- [47] C. Qingchao, L. Yang, W. Zhaowen, W. Ian, and C. Kevin, "Re-weighted adversarial adaptation network for unsupervised domain adaptation," in *Proc. CVPR*, 2018, pp. 7976–7985.
- [48] K. Saito, K. Watanabe, Y. Ushiku, and T. Harada, "Maximum classifier discrepancy for unsupervised domain adaptation," in *Proc. CVPR*, 2018, pp. 3723–3732.
- [49] C.-Y. Lee, T. Batra, M. H. Baig, and D. Ulbricht, "Sliced wasserstein discrepancy for unsupervised domain adaptation," in *Proc. CVPR*, 2019, pp. 10285–10295.
- [50] K. Bousmalis, N. Silberman, D. Dohan, D. Erhan, and D. Krishnan, "Unsupervised pixel-level domain adaptation with generative adversarial networks," in *Proc. CVPR*, 2017, pp. 3722–3731.
- [51] M.-Y. Liu and O. Tuzel, "Coupled generative adversarial networks," in *Proc. NIPS*, 2016, pp. 469–477.
- [52] Y. Taigman, A. Polyak, and L. Wolf, "Unsupervised cross-domain image generation," in *Proc. ICLR*, 2016, pp. 1–15.
- [53] S. Sankaranarayanan, Y. Balaji, C. Castillo, and R. Chellappa, "Generate to adapt: Aligning domains using generative adversarial networks," in *Proc. CVPR*, 2018, pp. 8503–8512.
- [54] J. Hoffman *et al.*, "CYCADA: Cycle-consistent adversarial domain adaptation," in *Proc. ICML*, 2018, pp. 1989–1998.
- [55] M.-Y. Liu, T. Breuel, and J. Kautz, "Unsupervised image-to-image translation networks," in *Proc. NIPS*, 2017, pp. 700–708.
- [56] G. Kang, L. Zheng, Y. Yan, and Y. Yang, "Deep adversarial attention alignment for unsupervised domain adaptation: The benefit of target expectation maximization," in *Proc. ECCV*, 2018, pp. 401–416.
- [57] P. Russo, F. M. Carlucci, T. Tommasi, and B. Caputo, "From source to target and back: Symmetric bi-directional adaptive gan," in *Proc. CVPR*, 2018, pp. 8099–8108.
- [58] Z. Murez, S. Kolouri, D. Kriegman, R. Ramamoorthi, and K. Kim, "Image to image translation for domain adaptation," in *Proc. CVPR*, 2018, pp. 4500–4509.
- [59] K. Bousmalis, G. Trigeorgis, N. Silberman, D. Krishnan, and D. Erhan, "Domain separation networks," in *Proc. NIPS*, 2016, pp. 343–351.
- [60] Y. Liu, X. Tian, Y. Li, Z. Xiong, and F. Wu, "Compact feature learning for multi-domain image classification," in *Proc. CVPR*, 2019, pp. 7193–7201.
- [61] G. Arthur, B. Karsten, R. Malte, S. Bernhard, and S. Alex, "A kernel two-sample test," *J. Mach. Learn. Res.*, vol. 13, pp. 723–773, Mar. 2012.
- [62] A. Gretton *et al.*, "Optimal kernel choice for large-scale two-sample tests," in *Proc. NIPS*, 2012, pp. 1205–1213.
- [63] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," in *Proc. ECCV*, vol. 9906, 2016, pp. 694–711.
- [64] J. Li, F. He, L. Zhang, B. Du, and D. Tao, "Progressive reconstruction of visual structure for image inpainting," in *Proc. ICCV*, 2019, pp. 5961–5970.
- [65] R. Olaf, F. Philipp, and B. Thomas, "U-Net: Convolutional networks for biomedical image segmentation," in *Proc. MICCAI*, 2015, pp. 234–241.
- [66] K. Saenko, B. Kulis, M. Fritz, and T. Darrell, "Adapting visual category models to new domains," in *Proc. ECCV*, 2010, pp. 213–226.
- [67] H. Venkateswara, J. Eusebio, S. Chakraborty, and S. Panchanathan, "Deep hashing network for unsupervised domain adaptation," in *Proc. CVPR*, 2017, pp. 5018–5027.
- [68] X. Peng, B. Usman, N. Kaushik, J. Hoffman, D. Wang, and K. Saenko, "VisDA: The visual domain adaptation challenge," 2017. [Online]. Available: arXiv:1710.06924
- [69] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.
- [70] J. J. Hull, "A database for handwritten text recognition research," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 16, no. 5, pp. 550–554, May 1994.
- [71] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng, "Reading digits in natural images with unsupervised feature learning," in *Proc. NIPS*, 2011, pp. 1–9.
- [72] T. Y. Lin *et al.*, "Microsoft COCO: Common objects in context," in *Proc. ECCV*, 2014, pp. 740–755.
- [73] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. CVPR*, 2016, pp. 770–778.
- [74] P. Adam *et al.*, "Automatic differentiation in pytorch," *Proc. NIPS*, 2017, pp. 1–4.
- [75] D. P. Kingma, and J. Ba, "Adam: A method for stochastic optimization," 2014. [Online]. Available: arXiv:1412.6980
- [76] M. Sugiyama, M. Krauledat, and K.-R. Müller, "Covariate shift adaptation by importance weighted cross validation," *J. Mach. Learn. Res.*, vol. 8, pp. 985–1005, Dec. 2007.
- [77] S. Roy, A. Siorohin, E. Sangineto, S. R. Bulò, N. Sebe, and E. Ricci, "Unsupervised domain adaptation using feature-whitening and consensus loss," in *Proc. CVPR*, 2020, pp. 9471–9480.
- [78] X. Chen, S. Wang, M. Long, and J. Wang, "Transferability vs. discriminability: Batch spectral penalization for adversarial domain adaptation," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 1081–1090.
- [79] P. O. Pinheiro, "Unsupervised domain adaptation with similarity learning," in *Proc. CVPR*, 2018, p. 8004–8013.
- [80] W. G. Chang, T. You, S. Seo, S. Kwak, and B. Han, "Domain-specific batch normalization for unsupervised domain adaptation," in *Proc. CVPR*, 2019, p. 7354–7362.
- [81] W. Zhang, W. Ouyang, W. Li, and D. Xu, "Collaborative and adversarial network for unsupervised domain adaptation," in *Proc. CVPR*, 2018, pp. 3801–3809.
- [82] G. French, M. Mackiewicz, and M. H. Fisher, "Self-ensembling for visual domain adaptation," in *Proc. ICLR*, 2018, pp. 1–15.

- [83] R. Xu, G. Li, J. Yang, and L. Lin, "Larger norm more transferable: An adaptive feature norm approach for unsupervised domain adaptation," in *Proc. ICCV*, 2019, pp. 1426–1435.
- [84] V. D. M. Laurens and G. Hinton, "Visualizing data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, p. 2579–2605, Nov. 2008.
- [85] Y. Mansour, M. Mohri, and A. Rostamizadeh, "Domain adaptation: Learning bounds and algorithms," in *Proc. COLT*, 2009, pp. 1–11.



Wanxia Deng received the university B.E. degree in electronic information science and technology from Xiamen University, Xiamen, China, in 2014, and the M.S. degree in information and communication engineering from the National University of Defense Technology, Changsha, China, in 2016, where she is currently pursuing the Ph.D. degree in information and communication engineering.

Her research interests include domain adaptation, transfer learning, deep learning, and image processing.



Lingjun Zhao received the B.S., M.S., and Ph.D. degrees in information and communication engineering from the National University of Defense Technology, Changsha, China, in 2003, 2004, and 2009, respectively.

She is currently an Associate Professor with the School of Electronic Science, National University of Defense Technology. Her research interest covers remote sensing information processing, SAR automatic target recognition, and deep learning.



Gangyao Kuang (Senior Member, IEEE) received the B.S. and M.S. degrees from the Central South University of Technology, Changsha, China, in 1988 and 1991, respectively, and the Ph.D. degree from the National University of Defense Technology, Changsha, in 1995.

He is currently a Professor with the School of Electronic Science, National University of Defense Technology. He was a Visiting Scholar with the University of Waterloo, Waterloo, ON, Canada, from 2009 to 2010. His current interests include remote

sensing, SAR image processing, change detection, SAR ground moving target indication, and the classification of polarimetric SAR images.



Dewen Hu (Senior Member, IEEE) received the B.S. and M.S. degrees from Xi'an Jiaotong University, Xi'an, China, in 1983 and 1986, respectively, and the Ph.D. degree from the National University of Defense Technology, Changsha, China, in 1999.

He is currently a Professor with the School of Intelligent Science, National University of Defense Technology. From 1995 to 1996, he was a Visiting Scholar with the University of Sheffield, Sheffield, U.K. His research interests include image processing, system identification and control, neural networks, and cognitive science.



Matti Pietikäinen (Fellow, IEEE) received the D.Sc. degree in technology from the University of Oulu, Oulu, Finland, in 1982.

He is an Emeritus Professor with the Center for Machine Vision and Signal Analysis, University of Oulu. From 1980 to 1981 and from 1984 to 1985, he visited the Computer Vision Laboratory, University of Maryland, College Park, MD, USA. He has authored more than 350 refereed papers in international journals, books, and conferences. His papers have about 57 000 citations in Google Scholar

(H-index 82). He has made fundamental contributions, e.g., to local binary pattern (LBP) methodology, texture-based image and video analysis, and facial image analysis.

Dr. Pietikäinen's research on LBP-based face description was awarded the Koenderink Prize for fundamental contributions in computer vision in 2014. He was the recipient of the IAPR King-Sun Fu Prize 2018 for fundamental contributions to texture analysis and facial image analysis. In 2018, he was named a Highly Cited researcher by Clarivate Analytics by producing multiple highly cited papers from 2006 to 2016, that rank in the top one percent by citation for his field in Web of Science. He was an Associate Editor of the IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, *Pattern Recognition*, the IEEE TRANSACTIONS ON FORENSICS AND SECURITY, and *Image and Vision Computing*. He currently serves as an Associate Editor for the IEEE TRANSACTIONS ON BIOMETRICS, BEHAVIOR AND IDENTITY SCIENCE, and a Guest Editor for special issues of the IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE and the *International Journal of Computer Vision*. He was a President of the Pattern Recognition Society of Finland from 1989 to 1992, and was named its honorary member in 2014. From 1989 to 2007, he served as a member of the Governing Board of International Association for Pattern Recognition (IAPR), and became one of the founding fellows of the IAPR in 1994. He is a Fellow of the IEEE for contributions to texture and facial image analysis for machine vision.



Li Liu (Senior Member, IEEE) received the Ph.D. degree in information and communication engineering from the National University of Defense Technology, China, in 2012.

She is currently a Professor with the College of System Engineering. During her Ph.D. study, she spent more than two years as a Visiting Student with the University of Waterloo, Canada, from 2008 to 2010. From 2015 to 2016, she spent ten months visiting the Multimedia Laboratory with the Chinese University of Hong Kong. From December 2016 to

November 2018, she worked as a Senior Researcher with the Machine Vision Group, the University of Oulu, Finland. Her current research interests include computer vision, pattern recognition, and machine learning. Her papers have currently over 3500+ citations in Google Scholar.