Dual-Correction–Adaptation Network for Noisy Knowledge Transfer
Yunyun Wang, Weiwen Zheng, Qinghao Li, and Songcan Chen, Senior Member, IEEE

Abstract—Unsupervised domain adaptation (UDA) promotes target learning via a single-directional transfer from label-rich source domain to unlabeled target, while its reverse adaptation from target to source has not been jointly considered yet. In real teaching practice, a teacher helps students learn and also gets promotion from students, and such a virtuous cycle inspires us to explore dual-directional transfer between domains. In fact, target pseudo-labels predicted by source commonly involve noise due to model bias; moreover, source domain usually contains innate noise, which inevitably aggravates target noise, leading to noise amplification. Transfer from target to source exploits target knowledge to rectify the adaptation, consequently enables better source transfer, and exploits a virtuous transfer circle. To this end, we propose a dual-correction–adaptation network (DualCAN), in which adaptation and correction cycle between domains, such that learning in both domains can be boosted gradually. To the best of our knowledge, this is the first naive attempt of dual-directional adaptation. Empirical results validate DualCAN with remarkable performance gains, particularly for extreme noisy tasks (e.g., approximately +10% on D→A of Office-31 with 40% label corruption).

Index Terms—Dual adaptation, feature noise, label noise, noise correction, unsupervised domain adaptation (UDA).

I. INTRODUCTION

A DEEP neural network has achieved remarkable success in many applications, such as computer vision [1], [2] and natural language processing [3], [4]. However, it relies on large-scare and high-quality annotated data, which is usually difficult to collect. Unsupervised domain adaptation (UDA) [5], [6], which aims to adopt a fully labeled source domain to help the learning of unlabeled target domain, has attracted much attention in recent years. Most UDA methods transfer knowledge from source to target by learning domain-invariant representation across domains, mainly including discrepancy-based [7], [8] and adversarial-based methods [9], [10], [11]. Discrepancy-based methods explicitly reduce the distribution discrepancy between domains by minimizing some distance metric, such as maximum mean discrepancy (MMD) [7], correlation alignment (CORAL) [12], and Wasserstein distance [8]. Based on generative adversarial learning methods [13], [14], align feature distributions across domains by adversarial training between feature generator and domain discriminator [9], or between different classifiers [15].

In real UDA tasks, the predicted target labels commonly involve noise due to model bias after source knowledge transfer. To prevent negative transfer from source domain, what and how to transfer [16], [17] are always the focus of domain adaptation. Moreover, the source domain usually involves noise as well, further giving rise to noisy UDA [18], [19]. For example, source data collected from crowd-sourced platforms or Internet media will inevitably be corrupted by label noise. Moreover, due to imaging equipment defects and network transmission compression, image instances often encounter pixel damage during data collection, leading to feature noise. The feature noise corrupts original features and aggravates the distribution discrepancy between source and target distributions, while label noise worsens the expected risk of classification, thus incurring misclassification of target instances. It makes previous UDA methods easy to fail in noisy environments. Recently, some studies [18], [19], [20], [21] have been dedicated to noisy UDA learning, which can be mainly divided into two categories. One kind of methods, such as transferable curriculum learning (TCL) [18] and robust domain adaptation (RDA) [19], adopts small-loss criterion to separate source instances into clean and noisy parts and then transfers source knowledge to target with only clean instances detected. The other kind, including noisy universal domain adaptation (Noisy UniDA) [20] and noise resistible mutual training (NRMT) [21], uses co-learning strategy with multiple classifiers to reduce the impact of label noise in adaptation.

Those previous UDA methods all adopt a single-directional knowledge transfer from labeled source to unlabeled target for helping the target learning. However, in some real teaching practice, a teacher helps students learn and also gets promotion from students to some extent. Inspired by such a virtuous-cyclical philosophy, the reverse adaptation from the target should intuitively be able to boost the source learning as well, especially for weak noisy sources. To the best of our knowledge, however, it has not been jointly considered in UDA so far. In fact, the target pseudo-labels predicted by source commonly involve noise due to transfer and model bias.

Manuscript received 3 November 2022; revised 11 May 2023; accepted 30 September 2023. Date of publication 19 October 2023; date of current version 8 January 2025. This work was supported in part by the NSFC under Grant 62076124 and Grant 62176118. (Corresponding author: Songcan Chen.)

Yunyun Wang, Weiwen Zheng, and Qinghao Li are with the Jiangsu Key Laboratory of Big Data Security and Intelligent Processing, Computer Science and Engineering, Nanjing University of Posts and Telecommunications, Nanjing 211106, China (e-mail: wangyunyun@njupt.edu.cn; 1020041209@njupt.edu.cn; 1021041212@njupt.edu.cn).

Songcan Chen is with the College of Computer Science and Technology, MIIT Key Laboratory of Pattern Analysis and Machine Intelligence, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China (e-mail: s.chen@nuaa.edu.cn).

Digital Object Identifier 10.1109/TNNLS.2023.32322390
At the same time, the source domain usually contains innate noise in real tasks. It will inevitably aggravate the target noise and incur noise amplification across domains. In this article, we attempt to explore a dual-directional knowledge transfer between domains and propose a dual-correction–adaptation network (DualCAN). In DualCAN, adaptation and correction cycle between domains, so as to achieve mutual promotion and cooperation across domains. Furthermore, a noise identification and correction (NIC) module is adopted to correct the noise in both domains. After that, those corrected instances are further recycled in learning rather than simply discarded, in order for a full knowledge utilization, especially in high noisy environment.

In implementation of DualCAN, knowledge transfer iterates back and forth between source-to-target (ST) task and target-to-source (TS) task. In ST, source knowledge is adapted to generate target pseudo-labels, and the pseudo-labels are further corrected by NIC with self-supervised knowledge. In TS, the target knowledge is transferred reversely to correct source noise and further boost source learning. With such a dual-directional knowledge transfer between domains, noise in both domains is corrected collaboratively, and performances in both domains are promoted mutually. It is analogous to the philosophy that teaching benefits both teacher and students alike. Quite naturally, dual-directional transfer can be adopted for both noisy and noise-free UDA tasks, while its learning concept is also applicable for some other related tasks, for example, using downstream tasks to reversely help feature learning in self-supervised learning. The main contributions of this article are summarized as follows.

1) A DualCAN is proposed for UDA learning. To the best of our knowledge, this is the first work of dual-directional adaptation to mutually promote learning and correct noise in both domains.

2) The noisy instances are corrected and recycled by a NIC module, in order to prevent noise amplification across domains and achieve a full knowledge utilization, especially in high noisy environment.

3) Empirical comparisons are conducted in real-world tasks under different noisy settings, in order to demonstrate the effectiveness of proposal.

The rest of this article is organized as follows. Section II introduces the related works, Section III gives the preliminaries, and the proposed DualCAN is described in detail in Section IV, and the comparison results are given in Section V. Finally, Section VI is the conclusion.

II. RELATED WORK

A. Domain Adaptation

In the early stage of deep domain adaptation, discrepancy-based UDA methods [7], [22] directly minimize the distribution discrepancy between domains to learn domain-invariant representations. The commonly adopted discrepancy metrics include MMD [7], CORAL [12], Wasserstein distance [8], and contrastive domain discrepancy (CDD) [23]. Recently, Wang et al. [24] induce an intermediate common representation space for both domains and match the embedding of data from both domains in the common representation space. Xia et al. [25] propose maximum structural generation discrepancy (MSGD) to estimate and mitigate domain shift by introducing an intermediate domain and generate class-consistent cross-domain instances in each mini-batch with the assistance of pseudo-labels. Inspired by the practice of adversarial learning, Ganin et al. [9] propose a domain adversarial neural network (DANN) to reduce domain gap in feature level. Since then, a series of studies have been proposed with adversarial learning. Liu and Tuzel [26] propose a generative adversarial network to learn joint distribution of multi-domain images. Long et al. [10] conduct an adversarial adaptation model using conditional distribution information. Hoffman et al. [27] perform adversarial learning from both pixel level and feature level for domain adaptation. Another line of work [28], [29], [30] treats domain adaptation as semi-supervised learning and adopts a self-training framework to boost knowledge transfer. Studies in [31], [32] have demonstrated the effectiveness of self-training in domain adaptation under reasonable assumptions. Liu et al. [33] improve standard self-training with cycle self-training (CST) with a two-head classifier sharing a feature extractor, so as to enforce pseudo-labels to generalize across domains. Since the target pseudo-labels predicted by source commonly involve noise due to transfer and model bias, there are also studies addressing the noise in target pseudo-labels. For example, Morerio et al. [34] adopt conditional generative adversarial networks to filter noise in pseudo-labels and generate cleaner target instances. Zheng and Yang [35] propose to rectify the target pseudo-labels via uncertainty estimation for domain adaptive semantic segmentation. However, they commonly adopt a single-directional knowledge transfer from source to target, while the adaptation from target to source has not been considered, which should be able to reversely boost source learning as well, especially for weak noisy sources.

B. Noisy UDA

Ambiguous features and incorrect labels seriously influence the generalization performance of deep CNNs. Previous methods address noise mainly by designing a robust loss function [36], [37], [38], [39] or filtering out noisy instances in the learning process [40], [41], [42], [43], [44]. When the noisy setting is introduced into UDA, the learning problem becomes much more complex, since the unreliability of target pseudo-labels will be incurred by not only domain discrepancy, but also source noise. To reduce the effect of noisy instances, one strategy follows the small-loss criterion to collect clean source data for adaptation. For example, Shu et al. [18] propose a transferable curriculum to enhance positive transfer from clean source instances, thus mitigating negative transfer by noise. Han et al. [19] improve the curriculum learning by retaining feature-corrupted data and use a proxy distribution in adversarial network. The other uses co-learning strategy with multiple classifiers to filter out source instances with incorrect annotations. For example, Zhao et al. [21] perform mutual instance selection to select reliable instances according to peer-confidence and relationship disagreement of networks. Yu et al. [20] study universal UDA in which target domain
UDA, i.e., distribution is commonly different from source distribution in X domains, which can also be applied in noise-free UDA tasks. Moreover, we use a two-way knowledge transfer and label noises in UDA, as well as the recycling of noisy labels in self-supervised learning.

Chen et al. [45] address noisy source-free UDA by fine-tuning between two classifiers to detect noisy source instances. Different from previous methods, we focus on both feature and label noises in UDA, as well as the recycling of noisy instances to achieve full knowledge utilization in high noisy environment. Moreover, we use a two-way knowledge transfer to mutually correct noise and promote performance in both domains, which can also be applied in noise-free UDA tasks.

III. PRELIMINARIES
A. Problem Statement
In UDA, both source instances \( X_S = \{x_i\}_{i=1}^{N_S} \) and labels \( Y_S = \{y_i\}_{i=1}^{N_S} \) are given, each instance \( x_i \in \mathbb{R}^d \) and \( y_i \in \{1, \ldots, K\} \), where \( N_S \) is the number of source instances, \( d \) is the feature dimension, and \( K \) is the class number. Target data \( X_T = \{x_i\}_{i=1}^{N_T} \) are an unlabeled set with \( N_T \) instances. The target distribution is commonly different from source distribution in UDA, i.e., \( p_S(x) \neq p_T(x) \) or \( p_S(x|y) \neq p_T(x|y) \). In real-world adaptation tasks, data collected are usually corrupted with both feature and label noises; thus, we usually encounter noisy source data. Specifically, assuming a noise ratio \( p_{\text{noise}} \), a clean instance \( x_i^{\text{CL}} \) will be corrupted by a noise \( e_i \) with a probability of \( p_{\text{noise}} \), i.e., \( p(x_i = x_i^{\text{CL}} + e_i) = p_{\text{noise}} \) and \( p(x_i = x_i^{\text{CL}}) = 1 - p_{\text{noise}} \). At the same time, clean label \( y_i^{\text{CL}} \) is corrupted according to a noise transition matrix \( T \in \mathbb{R}^{K \times K} \), where \( T_{il} = p(y_i = l|y_i^{\text{CL}} = k) = p_{\text{noise}} \) denotes the probability that instance \( x_i \) in the \( k \)th class is incorrectly labeled as \( l \).

B. Overall Concept
Previous UDA learning methods commonly adopt a single-directional adaptation, i.e., transfer knowledge only from source domain to target to boost target learning. Noisy UDA methods aim to filter out the noisy instances by some criterion, such as the small loss metric, or prediction discrepancy between different classifiers. Then, only clean or trusted source instances \( \{\tilde{X}_S, \tilde{Y}_S\} \) are selected for adaptation across domains, as shown in Fig. 1(a). While in this article, we propose dual-directional transfer and correction between domains, which contains unknown classes and optimize the divergence between two classifiers to detect noisy source instances. DualCAN is designed for UDA with both feature and label noises. It learns with dual-knowledge adaptation and noise correction between domains. We first introduce the network architecture of DualCAN and then describe the dual correction and adaptation process, as well as the NIC module for noise identification and correction in separate subsections, respectively.

IV. PROPOSED METHODS
DualCAN is designed for UDA with both feature and label noises. It learns with dual-knowledge adaptation and noise correction between domains. We first introduce the network architecture of DualCAN and then describe the dual correction and adaptation process, as well as the NIC module for noise identification and correction in separate subsections, respectively.

A. Network Architecture of DualCAN
The network architecture of DualCAN is shown in Fig. 2. Specifically, the source and target domains share the same feature generator \( G \) while different classifiers denoted as \( \theta_S \) and \( \theta_T \), respectively. First, noisy source data \( \{\tilde{X}_S, \tilde{Y}_S\} \) and target data \( \tilde{X}_T \) are set into the network. The initial feature generator and source classifier are obtained by source classification. Then, in the ST task, source knowledge is transferred to target by assigning the target pseudo-labels with the source classifier, and those pseudo-labels are further corrected by the NIC module to generate \( \hat{Y}_T \). The corrected
target pseudo-labels, along with self-supervised knowledge over target domain, are adopted to train the target classifier. Reversely, in the TS task, both source features and labels are corrected by NIC; then, feature generator $G$ is updated, so that both source and target classifiers can correctly classify the corrected source data $\{S, Y_s\}$; in this way, target knowledge is transferred to source to generate domain-invariant representation. Finally, a dual adaptation and correction framework is formed, so as to mutually transfer knowledge and correct noise across domains. Furthermore, in NIC, there are actually three steps. First, the small-loss criterion is adopted to roughly separate data into trusted and untrusted parts. Second, a semi-supervised clustering method is adopted to identify the feature and label noises in the untrusted part. Finally, those noises are further corrected for full knowledge utilization.

B. Dual Correction and Adaptation

In DualCAN, there is dual-directional knowledge transfer across domains. That is, knowledge transfer iterates between the forward ST task and backward TS task, so as to mutually promote learning in each domain.

The initial feature generator and source classifier are obtained by source classification. Then, in the forward ST task, knowledge transfer is conducted from label-rich source domain to the label-scarce target domain, or more specifically, target pseudo-labels $\tilde{Y}_T = \{\tilde{y}_i\}_i^{N_T}$ are first predicted by source classifier $\theta_S$, i.e.,

$$\tilde{y}_i = \arg\max_k x_i \sim X \{f_G, \theta_S(x_i)|x_i\} \tag{1}$$

where $f_G, \theta_S(x_i)|x_i$ is the $k$th element of target prediction. Those pseudo-labels actually contain inaccurate labels or noise due to cross-domain discrepancy and source noise as well. As a result, the values of $\tilde{Y}_T$ are further corrected by the NIC module, generating $\hat{Y}_T$ with improved label quality. Next, the target classifier $\theta_T$ is trained with the corrected target pseudo-labels, as well as self-supervised knowledge over target domain. Specifically, for each unlabeled target instance, we conduct a consistent loss between predictions for its weakly augmented version $x_i^1$ and a strongly augmented version $x_i^2$; finally, the objective for learning target classifier is written as follows:

$$\min_{\theta_T} \{L_{ce}(f_G, \theta_S(x_i), \tilde{y}_i) + L_{ce}(f_G, \theta_S(x_i^1), f_G, \theta_S(x_i^2)) \} \tag{2}$$

where $L_{ce}$ is the cross-entropy loss.

In the backward TS task, the learning goal is to transfer knowledge from corrected target domain to weak noisy source. First, both features and labels of untrusted source data are corrected by NIC, generating $\{\hat{X}_S, \hat{Y}_s\}$ to make it closer to a noise-free distribution. Then, with the target classifier $\theta_T$ fixed, the feature generator $G$ and source classifier $\theta_S$ are updated, such that both source and target classifiers can correctly classify source instances $\{S, Y_s\}$, in order to boost the target classifier on source domain to generate domain-invariant representation. The final learning objective is

$$\min_{G, \theta_S} \left\{L_{ce}(f_G, \theta_S(x_i), \hat{y}_i) \right. + \left. L_{ce}(f_G, \theta_S(x_i^1), f_G, \theta_S(x_i^2)) \right\} \tag{3}$$

Finally, the learning algorithm of DualCAN is shown in Algorithm 1, and the NIC module will be described in detail in Section IV-C. In DualCAN, the feature extractor is shared across domains. In fact, the feature extractor actually generates domain-invariant features, such that it retains the discriminative knowledge from source domain, while it keeps the self-supervised knowledge of target domain at the same time.

C. Noise Identification and Correction

Due to domain discrepancy and source noise, there will be an estimation bias in the predicted target pseudo-labels, which can also be considered as noisy labels. The target noise may hurt the source performance in the reverse adaptation. Moreover, there are actually both feature and label noises in the source domain, and it will aggravate the target noise, further leading to noise amplification across domains. Thus, in DualCAN, there is an extra NIC module for NIC in each iteration. It aims to detect untrusted data, identify them as feature or label noise, and then correct them to achieve full knowledge utilization. NIC actually learns in three steps.

In the first step, both source and target instances $\{X_{sT}, Y_{sT}\}$ are roughly divided into trusted data $[X_{sT}, \hat{Y}_{sT}]$ and untrusted data $[X_{sT}, \tilde{Y}_{sT}]$ by small-loss criterion, considering that clean instances have smaller losses than noisy instances, i.e.,

$$x_i \in \hat{X}_{sT}, \text{ if } L_{ce}(f_G, \theta_S(x_i), y_i) \leq \gamma \text{ [4]} \text{ x_i } \in X_{sT}, \text{ otherwise}$$

where $\gamma$ is the predefined threshold. If the loss of $x_i$ is smaller than $\gamma$, it will be treated as a clean instance and a noisy

| Algorithm 1 DualCAN |
|---------------------|
| **Input:** Noisy source data $\{X_s, Y_s\}$ and target data $X_T$ |
| **Output:** Feature extractor $G$; Source classifier $\theta_S$; Target classifier $\theta_T$ |

1. for epoch $= 0$ to $MaxEpoch$
   2. /* forward source to target task */
      3. Generate pseudo-labels $\hat{Y}_T$ on $X_T$ with $G$ and $\theta_S$ by Eq. 1.
      4. Correct target pseudo-labels $\hat{Y}_T$ by the NIC module to generate $\tilde{Y}_T$.
      5. Train target classifier $\theta_T$ with corrected pseudo-label $\tilde{Y}_T$ and self-supervised consistency by Eq.2.
   6. /* backward target to source task */
      7. Correct both feature and label noises in noisy source dataset $\{X_s, Y_s\}$ by NIC module to generate $\{\hat{X}_s, \hat{Y}_s\}$.
   8. Update feature extractor $G$ and source classifier $\theta_S$ by Eq.3.
   9. end for
one otherwise. In fact, only a small fraction of instances are selected into the trusted set in this step, in order to avoid introducing noise into a clean dataset. That is, the untrusted set still contains both clean instances and noise. For a source instance \( x_i, y_i \) is its given class label, and it is classified by source classifier to get the classification loss, i.e., \( f_{G, \theta_S}(x_i) = \theta_S(G(x_i)) \). While for target instance \( x_i, y_i \) is the predicted pseudo-label by source classifier, and \( x_i \) is classified by target classifier, i.e., \( f_{G, \theta_T}(x_i) = \theta_T(G(x_i)) \). Since there is no ground truth for target instances, we actually consider that if its prediction by target classifier is inconsistent with the pseudo-label from source classifier, then the pseudo-label can be unreliable and, thus, be viewed as a noisy label.

In the second step, since the trusted set contains both clean instances and noise, including feature and label noises, those different kinds of instances are further identified to reduce the negative effect. Referring to semi-supervised learning, a deep clustering method [46] is adopted here, which can separate mixed noise through increasing the divergence between feature noise of outliers and label noise. Specifically, the trusted instances by the previous step are viewed as labeled data, while the untrusted instances are treated as unlabeled data. \( K \) initial clusters \( C_k (k = 1, \ldots , K) \) are first formed over trusted data in terms of their class labels, i.e., the \( k \)th cluster consists of \( N_k \) trusted instances that belong to the \( k \)th class

\[
C_k = \{(x_i, y_i) | y_i = k\}^\mathcal{N}_k. \tag{5}
\]

The corresponding cluster center \( \mu_k \) is computed as the average of instances in each cluster

\[
\mu_k = \frac{1}{|C_k|} \sum_{x_i \in C_k} G(x_i) \tag{6}
\]

and the cluster radius \( r_k \) is the Euclidean distance between the farthest trusted instance and cluster center

\[
r_k = \max_{x_i \in C_k} \|G(x_i) - \mu_k\|_2. \tag{7}
\]

Then, for an untrusted instance \( \tilde{x}_i \), its cluster index is assigned by the nearest cluster, i.e.,

\[
k_i^* = \arg \min_{k \in \{1, 2, \ldots , K\}} (\|G(\tilde{x}_i) - \mu_k\|_2). \tag{8}
\]

For a source instance \( \tilde{x}_i \) in the untrusted set, it will be identified as follows:

\[
\tilde{x}_i \in \begin{cases} X^F_S, & \text{if } \|G(\tilde{x}_i) - \mu_{k_i^*}\|_2 > r_{k_i^*} \\ X^L_S, & \text{if } \|G(\tilde{x}_i) - \mu_{k_i^*}\|_2 \leq r_{k_i^*} \text{ and } \tilde{y}_i \neq k_i^* \\ X^C_S, & \text{otherwise} \end{cases} \tag{9}
\]

where \( X^F_S, X^L_S \), and \( X^C_S \) denote the feature noise, label noise, and clean source sets, respectively. Specifically, if \( \tilde{x}_i \) is outside the radius of the nearest cluster \( C_{k_i^*} \), then \( \tilde{x}_i \) has no obvious common features with the trusted data; thus, it is identified as feature noise. If \( \tilde{x}_i \) is within the radius but its label is different from the cluster center, then it has high feature similarity with trusted data but a wrong label; finally, it is identified as label noise. The other instances not covered in the above two groups are assigned to clean source set with normal features and labels. For a target instance in untrusted set, its predicted pseudo-label will be identified as a noisy label directly.

Finally, those noisy instances are corrected after identification. In some cases, adding feature disturbance can help increase the classifier robustness, while severe feature noise will increase the difficulty of representation learning and incur wrong predictions. To utilize the useful part in feature noise, we take the class center \( \mu_{k_i^*} \) as disturbance feature and perform weighted disturbance on noisy features to correct instances at the feature level, i.e.,

\[
G(\tilde{x}_i) = (1 - \eta)G(\tilde{x}_i) + \eta \mu_{k_i^*} \tag{10}
\]

where the disturbance weight \( 0 \leq \eta \leq 1 \) controls the invasion proportion of disturbance. In practice, we dynamically adjust it during training, that is, gradually reduce \( \eta \) to zero as training continues. In the beginning, feature noise is corrected for better domain alignment, while as training proceeds, the noise becomes weaker, which can be adopted to improve the model robustness. For label noise, the corrected label \( \tilde{y}_i \) is directly specified by the nearest cluster label \( k_i^* \), i.e., \( \tilde{y}_i = k_i^* \).

The whole learning process of the NIC module is summarized in Algorithm 2.

### D. Theory Justification

In this section, we give a theory justification for DualCAN, in terms of a noisy version of generalization bound for target domain.

**Proposition 1** [19]: For any hypothesis \( h \in \mathcal{H} \), the bound of target expected risk \( \epsilon_T(h) \) in noisy environments is given by

\[
\epsilon_T(h) \leq \epsilon_S(h) + |\epsilon_S(h, h^*) - \epsilon_T(h, h^*)| + \lambda \tag{11}
\]
where \( \lambda = \varepsilon_S(h^*, f_S) + \varepsilon_T(h^*, f_T) \) is the ideal combined error of \( h^*, f_S \) and \( f_T \) are the true labeling functions for source and target domains, respectively, and

\[
h^* = \arg \min_{h \in \mathcal{H}} \varepsilon_S(h, f_S) + \varepsilon_T(h, f_T).
\]

(12)

From proposition 1, the target risk in noisy UDA is bounded by three items. The first item is the empirical risk of noisy source data. In DualCAN, a reverse knowledge transfer from target to source is adopted, and the aligned target data can actually be treated as augmented source data after adaptation; thus, the target data structure can help correct source noise and reduce source risk. At the same time, noisy source instances are corrected and recycled in DualCAN, which can further reduce the source risk, especially under extremely noisy environment. In fact, a better source learning guarantees better target learning, and reversely, it helps correct source noise and promote source performance as well; finally, a virtuous circle is formed. The second item is the distribution discrepancy across noisy source domain and target domain, and the feature noise in source domain will affect the distribution discrepancy [19]. Considering that, feature noise in source domain is also addressed in DualCAN for better domain alignment.

Furthermore, for any labeling functions \( f_1 - f_3 \), we have [47]

\[
\varepsilon(f_1, f_2) \leq \varepsilon(f_1, f_3) + \varepsilon(f_2, f_3).
\]

Then, we have

\[
\lambda = \min_{h \in \mathcal{H}} \varepsilon_S(h, f_S) + \varepsilon_T(h, f_T) \\
\leq \min_{h \in \mathcal{H}} \varepsilon_S(h, f_S) + \varepsilon_T(h, f_S) + \varepsilon_T(f_S, f_T).
\]

(13)

The first and second items denote the disagreement between \( h \) and the source labeling function \( f_S \) on source and target data, respectively. The third item is the gap of source and target functions over target instances, which is actually fixed with given data. Since the source instances are labeled, the first item can be very small. As a result, we focus on the second item \( \varepsilon_T(h, f_S) = \sum_{x \in X_S} [l(h(x), f_S(x)) - l(h(x))] \) for smaller \( \lambda \), where \( l(\cdot, \cdot) \) is the 0–1 loss. By reversely transferring knowledge from target to source, it is helpful to reduce the discrepancy between \( h \) and \( f_S \) over the target data.

In summary, by dual-directional knowledge transfer between domains, as well as correction and recycling for both feature and label noises, it can be expected to achieve better performance by DualCAN.

V. EXPERIMENTS

In this section, we evaluate the proposed DualCAN on three vision datasets, compared with the state-of-the-art UDA and noisy UDA methods.

A. Setup

1) Datasets: Office-31 [48] is a standard domain adaptation dataset containing 4652 images with 31 classes. It consists of three domains: Amazon (A), Webcam (W), and DSLR (D). Office-Home [49] has 15,599 images with 65 classes. It contains four domains with large domain gaps: Artistic (Ar), Clip Art (Cl), Product (Pr), and Real World (Rw). To introduce noise in those two clean datasets, we follow the protocol in [19] to create corrupted counterparts in three different ways: label corruption, feature corruption, and mixed corruption. Label corruption changes the label of each image to other random classes uniformly and equally with probability \( p_{\text{noise}} \). We use salt-and-pepper noise and Gaussian filter to simulate the imaging noise and blur generated by image compression process, respectively. Then, feature corruption corrupts pixels of each image with a probability of \( p_{\text{noise}} \) by Gaussian blur and salt-and-pepper noise. Mixed corruption refers that each image is corrupted by label corruption and feature corruption with probability \( p_{\text{noise}}^2 \) independently. Bing-Caltech [50] is a real noisy dataset consisting of Bing (B) and Caltech-256 (C). The Bing dataset is created by collecting images from the Bing image search engine with class labels in Caltech-256, which naturally contains label and feature noises. We take Bing as the noisy source domain and Caltech-256 as the clean target domain.

2) Benchmark Methods: We compare DualCAN with state-of-the-art methods: ResNet-50 [51], self-paced learning (SPL) [52], MentorNet [42], deep adaptation network (DAN) [7], residual transfer network (RTN) [53], DANN [9], conditional adversarial domain adaptation network (CDAN) [10], margin disparity discrepancy (MDD) [54], CST [33], TCL [18], and RDA [19]. SPL and MentorNet are label noise processing methods, and TCL and RDA are noisy domain adaptation methods, while the others are standard UDA methods.

3) Implementation: We use all labeled source and unlabeled target instances for training following the standard protocols in UDA [9] and implement both DualCAN and comparison methods in PyTorch. We use ResNet-50 pretrained on ImageNet [55] as feature extractors and a fully connected bottleneck layer before the classification layer. We set the sample selection pretraining epoch to 30. The small-loss threshold \( \gamma \) is usually determined by empirical or prior knowledge of noise in previous methods [18], [19]. We fix the separation ratio \( p = 0.08 \) to set \( \gamma \) as the loss of the (\( N \times p \))th instance in most tasks. Following standard protocol in [51], we set the initial learning rate to \( 2e^{-3} \) and decay the learning rate by 0.1 in each 30 of the 90 epochs.

B. Results

1) Noisy UDA: Table I shows the results on Office-31 under 40% label corruption, feature corruption, and mixed corruption, respectively. In mixed corruption case, 40% means 20% feature corruption and 20% label corruption independently. In each column, the bold value indicates the best performance among compared methods, and the value with underline is the second-best performance. From those tables, we can find that the following hold.

1) Through addressing label noise in learning, both SPL and MentorNet obtain better performance than ResNet; however, since they directly apply source classifier over target instances without taking the domain discrepancy into account, their target performance can be further boosted.
2) The UDA methods, such as MDD, only consider the domain alignment in learning, whereas there is noise in source domain, and the target learning will be affected by the source noise, leading to negative transfer. As a result, their performance can be further boosted, and it is important to correct the source noise in domain adaptation for better knowledge transfer.

3) TCL and RDA consider both source noise and domain discrepancy in learning. They filter out noise instances with small-loss criterion to reduce the negative impact of noise in adaptation, thus achieving better performance than those methods considering noise processing or domain discrepancy alone.

4) Our DualCAN outperforms the other methods in almost all cases, including TCL and RDA, indicating the effectiveness of the proposed dual correction and adaptation.

Table II shows the results on Office-Home with 40% mixed corruption, including 20% label corruption and 20% feature corruption independently, as well as naive noisy environment over Bing-Caltech. From this table, it can be found that DualCAN achieves competitive performance compared with the state-of-art UDA methods. Specifically, DualCAN obtains the best performance over five out of the nine adaptation tasks over Office-31 and over nine out of the 12 tasks over Office-Home. Moreover, it achieves the best average performance on both datasets. As a result, a dual-directional knowledge transfer across domains achieves competitive performance in noise-free UDA as well.

3) Source Performance: In DualCAN, performance in both domains can be promoted; thus, we also show the average source performance over the six tasks of Office-31 under different noise ratios in Fig. 3, although the target performance is actually our focus. From Fig. 3, we can find that DualCAN achieves better source performance than the second-best

Fig. 3. Comparison of average source performance with different noise levels over Office-31.
Fig. 4. (a) Loss distributions of instances. (b) Distance distribution of instances by clustering. (c) Visualization of deep clusters for random three classes in Office-Home Product. The green dashed circles represent the noise determination range with green cluster centers and $r_k$ as radius.

**TABLE III**

| Method | Ar→Cl | Ar→Pr | Ar→Rw | Cl→Ar | Cl→Pr | Cl→Rw | Pr→Ar | Pr→Cl | Pr→Rw | Rw→Ar | Rw→Cl | Rw→Pr | Avg |
|--------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-----|
| ResNet | 34.9  | 50.0  | 58.0  | 37.4  | 41.9  | 46.2  | 38.5  | 31.2  | 60.4  | 53.9  | 51.2  | 59.9  | 46.1|
| DAN    | 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|
| DANN   | 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|
| CDAN   | 50.7  | 70.6  | 76.0  | 57.0  | 70.0  | 70.0  | 57.4  | 50.9  | 77.3  | 70.9  | 56.7  | 81.6  | 65.8|
| MDD    | 54.9  | 73.7  | 77.8  | 60.0  | 71.4  | 71.8  | 61.2  | 53.6  | 78.1  | 72.5  | 60.2  | 82.3  | 68.1|
| CST    | 59.0  | 79.6  | **83.4** | 68.4  | 77.1  | 76.7  | 68.9  | 56.4  | **83.0** | 75.3  | 62.2  | **85.1** | 73.0|
| DualCAN | **61.7** | **79.8** | **83.1** | **69.4** | **79.8** | **77.1** | **70.5** | **57.2** | **81.7** | **77.9** | **62.5** | **83.9** | **73.7** |

Fig. 5. Analysis of various methods in different mixed noise levels.

Fig. 6. Source noise ratio and target pseudo-label error rate after correction with respect to training epochs.

method RDA, and the reason can be that DualCAN uses target knowledge to correct the source noise, while RDA only detects the noise and then learns with just trusted instances. With dual correction and adaptation in DualCAN, noise in both domains can be corrected mutually, forming a virtuous cycle of adaptation.

**C. Analysis**

1) **Ablation Study**: In order to investigate the contribution of each component in DualCAN better, we perform an ablation study on Office-31 with 40% mixed corruption by discarding some components in DualCAN, and the results are reported in Table V. From Table V, it can be found that removing the NIC module in DualCAN degenerates the target performance; thus, it is helpful to correct noise with NIC in noisy UDA learning. Second, discarding either label or feature correction for source domain leads to performance degradation; as a result, it is reasonable to address both label and feature noises in learning, while label noise commonly has a greater impact on the performance. Finally, the performance of DualCAN is better than source correction or target correction alone, which indicates that dual correction is useful for alleviating the negative effects of noisy instances and pseudo-labels and, thus, helps boost both source and target learning. Besides, the NIC module is also validated in noise-free UDA setting, where there is no source noise, but pseudo-label noise in target domain. Specifically, the performance of DualCAN without NIC is added in the penultimate row of Table IV, from which we can find that DualCAN with NIC achieves better performance; as a result, NIC can help correct the noise lying in target pseudo-labels and, thus, is helpful for noise-free UDA learning as well.

2) **Noise Distribution**: In DualCAN, a semi-supervised deep clustering is adopted to identify feature and label noises.
In Fig. 4, we show the separability of feature or label noise from clean instances on Product domain of Office-Home. Specifically, the separability by traditional loss distribution in RDA is shown in Fig. 4(a), and the separability by clustering in DualCAN is shown in Fig. 4(b), that is, the separability by distribution of distances from instances to the nearest cluster center. In Fig. 4(a), there are large overlaps between feature and label noises, while the distance by clustering can better identify feature and label noises, as shown in Fig. 4(b). Furthermore, we visualize clusters of three random classes in Fig. 4(c), in which different shapes denote instances in different classes and red triangles denote label-noisy instances that are incorrectly labeled as the other 62 categories. From this figure, it can easily be found that instances with label noise are commonly closer to cluster centers than the ones with feature noise, and there are clear boundaries among clusters. It verifies that with clustering, our proposed NIC can well identify feature and label noises for further noise correction.

3) Noise Levels: Fig. 5 reports the performance of individual methods in a wide range of mixed noise levels on A→W task of Office-31. Specifically, the noise level is from 0.0 to 1.6, where 0.0 means the noise-free UDA scenario and 1.6 denotes 160% mix corruption with 80% feature corruption and 80% feature corruption independently. From Fig. 5, as the noise level increases, the performance decreases for all methods, especially for DANN and ResNet with no consideration of noise processing. The performance of DualCAN is better than the other methods and more stable with noise as well. It is noted that when the noise level is 1.6, DualCAN performs much better than other methods, including TCL and RDA. The reason can be that DualCAN corrects and recycles noise in learning and, thus, can make full use of data, especially in high noisy environment. At the same time, DualCAN also achieves the best performance when the noise level is 0, which proves that our proposal is applicable in standard UDA scenarios.

4) Corrected Sample Quality: In order to show the quality of noise correction, we visualize the source noise ratio and the target pseudo-label error, respectively, on Pr→Rw task of Office-Home in Fig. 6. As the training proceeds, the proportion of source noise gradually decreases, and the error of target pseudo-label decays and achieves stable in a few epochs.

5) Feature Visualization: Fig. 7 exhibits the t-distributed stochastic neighbor embedding (t-SNE) [56] of the bottleneck representation by DANN, TCL, RDA, and DualCAN, respectively, with 40% mixed corruption on Pr→Rw of Office-Home. Specifically, Fig. 7(a)–(c) shows the learned target features with different colors indicating different classes,
while Fig. 7(e)–(h) displays the learned target features with different colors indicating different domains. From Fig. 7(a)–(c), the learned features from DANN are mixed up, while DualCAN distinguishes categories better than the other compared methods, including both TCL and RDA; thus, it is more robust to noise. At the same time, in Fig. 7(e)–(h), the source and target domains are better aligned by DualCAN. Moreover, DualCAN obtains clearer boundaries, better within-class compactness, and less outliers. As a result, DualCAN provides a better solution for noisy UDA.

6) Time Complexity: The training time comparison between DualCAN and CST over Office-31 is given in Fig. 8, since both DualCAN and CST adopt the self-training strategy for UDA learning. We use ResNet-50 as the backbone architecture over devices configured as NVIDIA GeForce RTX 3090 graphics cards. From Fig. 8, the training cost of CST is lower than that of DualCAN, since CST does not handle the noise in both source and target domain. However, considering the learning performance, it is still worth to correct the noise with DualCAN at the expense of training time.

VI. CONCLUSION

Previous UDA methods commonly focus on the single-directional ST knowledge transfer, while in this article, a DualCAN is proposed with dual-directional knowledge transfer and noise correction across domains. In DualCAN, knowledge transfer iterates between ST task and TS task. In ST, target pseudo-labels are generated and corrected in terms of source knowledge. While in TS, target knowledge is transferred backward to correct the source noise. Moreover, both feature and label noises are addressed in DualCAN, and those noisy instances are corrected and recycled by an NIC module. Empirical results indicate that DualCAN achieves remarkable performance improvement over prior state of the arts.

REFERENCES

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” Commun. ACM, vol. 60, no. 6, pp. 84–90, May 2017.
[2] X. Chang, P. Ren, P. Xu, Z. Li, X. Chen, and A. Hauptmann, “A comprehensive survey of scene graphs: Generation and application,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 45, no. 1, pp. 1–26, Jan. 2023.
[3] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of deep bidirectional transformers for language understanding,” 2018, arXiv:1810.04805.
[4] M. Li, P.-Y. Huang, X. Chang, J. Hu, Y. Yang, and A. Hauptmann, “Video pivoting unsupervised multi-modal machine translation,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 45, no. 3, pp. 3918–3932, Mar. 2023.
[5] 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, May 2010.
[6] S. Zhao et al., “A review of single-source deep unsupervised visual domain adaptation,” IEEE Trans. Neural Netw. Learn. Syst., vol. 33, no. 2, pp. 473–493, Feb. 2022.
[7] M. Long, Y. Cao, J. Wang, and M. Jordan, “Learning transferable features with deep adaptation networks,” in Proc. Int. Conf. Mach. Learn., 2015, pp. 97–105.
[8] J. Shen, Y. Qu, W. Zhang, and Y. Yu, “Wasserstein distance guided representation learning for domain adaptation,” in Proc. AAAI Conf. Artif. Intell., vol. 32, no. 1, Apr. 2018, pp. 1–8.
[9] Y. Ganin et al., “Domain-adversarial training of neural networks,” J. Mach. Learn. Res., vol. 17, no. 1, pp. 2030–2096, 2017.
[10] M. Long, Z. Cao, J. Wang, and M. I. Jordan, “Conditional adversarial domain adaptation,” in Proc. Adv. Neural Inf. Process. Syst., vol. 31, 2018, pp. 1–11.
[11] L. Li, M.-W. Mak, and J.-T. Chien, “Contrastive adversarial domain adaptation networks for speaker recognition,” IEEE Trans. Neural Netw. Learn. Syst., vol. 33, no. 5, pp. 2236–2245, May 2022.
[12] B. Sun, J. Feng, and K. Saenko, “Return of frustratingly easy domain adaptation,” in Proc. AAAI Conf. Artif. Intell., vol. 30, no. 1, Mar. 2016, pp. 1–8.
[13] I. Goodfellow et al., “Generative adversarial networks,” Commun. ACM, vol. 63, no. 11, pp. 139–144, 2020.
[14] C. Yan et al., “ZeroNAS: Differentiable generative adversarial networks search for zero-shot learning,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 44, no. 12, pp. 9733–9740, Dec. 2022.
[15] K. Saito, K. Watanabe, Y. Ushiku, and T. Harada, “Maximum classifier discrepancy for unsupervised domain adaptation,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., Jun. 2018, pp. 3723–3732.
[16] S. J. Pan and Q. Yang, “A survey on transfer learning,” IEEE Trans. Knowl. Data Eng., vol. 22, no. 10, pp. 1345–1359, Oct. 2010.
[17] L. Zhang et al., “TN-ZSTAD: Transferable network for zero-shot temporal activity detection,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 45, no. 3, pp. 3848–3861, Mar. 2023.
[18] Y. Shu, Z. Cao, M. Long, and J. Wang, “Transferable curriculum for weakly-supervised domain adaptation,” in Proc. AAAI Conf. Artif. Intell., Jul. 2019, vol. 33, no. 1, pp. 4971–4983.
[19] Z. Han, X.-J. Gui, C. Cui, and Y. Yin, “Towards accurate and robust domain adaptation under noisy environments,” 2020, arXiv:2004.12529.
[20] Q. Yu, A. Hashimoto, and Y. Ushiku, “Divergence optimization for noisy universal domain adaptation,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2021, pp. 2515–2524.
[21] F. Zhao, S. Liao, G.-S. Xie, J. Zhao, K. Zhang, and L. Shao, “Unsupervised domain adaptation with noise resistant mutual-training for person re-identification,” in Proc. Eur. Conf. Comput. Vis. Cham, Switzerland: Springer, 2020, pp. 526–544.
[22] W. Zellinger, T. Grubinger, E. Lughofer, T. Natschläger, and J. W. Vaughan, “A theory of learning from different domains,” in Proc. Adv. Neural Inf. Process. Syst., vol. 31, 2018, pp. 1–11.
[23] G. Kang, L. Jiang, Y. Yang, and A. G. Hauptmann, “Contrastive adaptation network for unsupervised domain adaptation,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2019, pp. 4893–4902.
[24] Z. Wang, B. Du, and Y. Guo, “Domain adaptation with neural embedding matching,” IEEE Trans. neural Netw. Learn. Syst., vol. 31, no. 7, pp. 2387–2397, Jul. 2019.
[25] H. Xia, T. Jing, and Z. Ding, “Maximum structural generation discrepancy for unsupervised domain adaptation,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 45, no. 3, pp. 3434–3445, Mar. 2023.
[26] M.-Y. Liu and O. Tuzel, “Coupled generative adversarial networks,” in Proc. Adv. Neural Inf. Process. Syst., vol. 29, 2016, pp. 1–9.
[27] J. Hoffman et al., “CyCADA: Cycle-consistent adversarial domain adaptation,” in Proc. Int. Conf. Mach. Learn., 2018, pp. 1989–1998.
[28] G. Frensch, M. Mackiewicz, and M. Fisher, “Self-ensembling for visual domain adaptation,” 2017, arXiv:1706.05208.
[29] D.-H. Lee et al., “Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks,” in Proc. Workshop Challenges Represent. Learn. (ICML), vol. 3, no. 2, 2013, p. 896.

[30] Y. Zou, Z. Yu, X. Liu, B. Kumar, and J. Wang, “Confidence regularized self-training,” in Proc. IEEE/CVF Int. Conf. Comput. Vis., Oct. 2019, pp. 5982–5991.

[31] A. Kumar, T. Ma, and P. Liang, “Understanding self-training for gradual domain adaptation,” in Proc. Int. Conf. Mach. Learn., 2020, pp. 5468–5479.

[32] T. Cai, R. Gao, J. Lee, and Q. Lei, “A theory of label propagation for subpopulation shift,” in Proc. Int. Conf. Mach. Learn., 2021, pp. 1170–1182.

[33] H. Liu, J. Wang, and M. Long, “Cycle self-training for domain adaptation,” in Proc. Adv. Neural Inf. Process. Syst., vol. 34, 2021, pp. 1–14.

[34] P. Morerio, R. Volpi, R. Ragonesi, and V. Murino, “Generative pseudo-label refinement for unsupervised domain adaptation,” in Proc. IEEE/CVF Winter Conf. Appl. Comput. Vis., Mar. 2020, pp. 3130–3139.

[35] Z. Zheng and Y. Yang, “Rectifying pseudo label learning via uncertainty estimation for domain adaptive semantic segmentation,” Int. J. Comput. Vis., vol. 129, no. 4, pp. 1106–1120, Apr. 2021.

[36] T. Miyato, S.-I. Maeda, M. Koyama, and S. Ishii, “Virtual adversarial training: A regularization method for supervised and semi-supervised learning,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 41, no. 8, pp. 1979–1993, Aug. 2018.

[37] G. Patrini, A. Rozza, A. K. Menon, R. Nock, and L. Qu, “Making deep neural networks robust to label noise: A loss correction approach,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., Jul. 2017, pp. 1944–1952.

[38] S. Reed, H. Lee, D. Anguelov, C. Szegedy, D. Erhan, and A. Rabinovich, “Training deep neural networks on noisy labels with bootstrapping,” 2014, arXiv:1412.6596.

[39] L. Liu, Q. Feng, C. L. P. Chen, and Y. Wang, “Noise robust face hallucination based on scene context representation,” IEEE Trans. Neural Netw. Learn. Syst., vol. 33, no. 10, pp. 5953–5965, Oct. 2022.

[40] B. Han et al., “Co-teaching: Robust training of deep neural networks with extremely noisy labels,” in Proc. Adv. Neural Inf. Process. Syst., vol. 31, 2018, pp. 1–11.

[41] J. Liu, R. Li, and C. Sun, “Co-correcting: Noise-tolerant medical image classification via mutual label correction,” IEEE Trans. Med. Imag., vol. 41, no. 12, pp. 3580–3592, Dec. 2021.

[42] L. Jiang, Z. Zhou, T. Leung, L.-J. Li, and L. Fei-Fei, “MentorNet: Learning data-driven curriculum for very deep neural networks on corrupted labels,” in Proc. Int. Conf. Mach. Learn., 2018, pp. 2304–2313.

[43] N. Natrajan, I. S. Dhillon, P. K. Ravikumar, and A. Tewari, “Learning with noisy labels,” in Proc. Adv. Neural Inf. Process. Syst., vol. 26, 2013, pp. 1–9.

[44] S. Sukhbaatar, J. Bruna, M. Paluri, L. Bourdev, and R. Fergus, “Training convolutional networks with noisy labels,” 2014, arXiv:1406.2080.

[45] W. Chen et al., “Self-supervised noisy label learning for source-free unsupervised domain adaptation,” 2021, arXiv:2102.11614.

[46] S. Park et al., “Improving unsupervised image clustering with robust learning,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., Jun. 2021, pp. 12278–12287.

[47] Y. Zhu et al., “Deep subdomain adaptation network for image classification,” IEEE Trans. Neural Netw. Learn. Syst., vol. 32, no. 4, pp. 1713–1722, Apr. 2021.

[48] K. Saenko, B. Kulis, M. Fritz, and T. Darrell, “Adapting visual category models to new domains,” in Proc. Eur. Confr. Comput. Vis. Cham, Switzerland: Springer, 2010, pp. 213–226.

[49] H. Venkateswara, J. Eusebio, S. Chakraborty, and S. Panchanathan, “Deep hashing network for unsupervised domain adaptation,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., Jul. 2017, pp. 5018–5027.

[50] A. Bergamo and L. Torresani, “Exploiting weakly-labeled web images to improve object classification: A domain adaptation approach,” in Proc. Adv. Neural Inf. Process. Syst., vol. 23, 2010, pp. 1–9.

[51] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., Jul. 2016, pp. 770–778.

[52] M. Kumar, B. Packer, and D. Koller, “Self-paced learning for latent variable models,” in Proc. Adv. Neural Inf. Process. Syst., vol. 23, 2010, pp. 1–9.

[53] M. Long, H. Zhu, J. Wang, and M. I. Jordan, “Unsupervised domain adaptation with residual transfer networks,” in Proc. Adv. Neural Inf. Process. Syst., vol. 29, 2016, pp. 1–9.

[54] Y. Zhang, T. Liu, M. Long, and M. Jordan, “Bridging theory and algorithm for domain adaptation,” in Proc. Int. Conf. Mach. Learn., 2019, pp. 7404–7413.

[55] O. Russakovsky et al., “ImageNet large scale visual recognition challenge,” Int. J. Comput. Vis., vol. 115, no. 3, pp. 211–252, Dec. 2015.

[56] J. Donahue et al., “DeCAF: A deep convolutional activation feature for generic visual recognition,” in Proc. Int. Conf. Mach. Learn., 2014, pp. 647–655.