Contrastive Adaptation Network for Single- and Multi-Source Domain Adaptation

Guoliang Kang, Lu Jiang, Yunchao Wei, Yi Yang, and Alexander Hauptmann

Abstract—Unsupervised domain adaptation (UDA) makes predictions for the target domain data while manual annotations are only available in the source domain. Previous methods minimize the domain discrepancy neglecting the class information, which may lead to misalignment and poor generalization performance. To tackle this issue, this paper proposes contrastive adaptation network (CAN) that optimizes a new metric named Contrastive Domain Discrepancy explicitly modeling the intra-class domain discrepancy and the inter-class domain discrepancy. To optimize CAN, two technical issues need to be addressed: 1) the target labels are not available; and 2) the conventional mini-batch sampling is imbalanced. Thus we design an alternating update strategy to optimize both the target label estimations and the feature representations. Moreover, we develop class-aware sampling to enable more efficient and effective training. Our framework can be generally applied to the single-source and multi-source domain adaptation scenarios. In particular, to deal with *multiple source domain data*, we propose: 1) *multi-source clustering ensemble* which exploits the complementary knowledge of distinct source domains to make more accurate and robust target label estimations; and 2) *boundary-sensitive alignment* to make the decision boundary better fitted to the target. Experiments are conducted on three real-world benchmarks (i.e., Office-31 and VisDA-2017 for the single-source scenario, DomainNet for the multi-source scenario). All the results demonstrate that our CAN performs favorably against the state-of-the-art methods. Ablation studies also verify the effectiveness of each key component of our proposed system.

Index Terms—Contrastive, domain adaptation, unsupervised, multi-source

1 Introduction

Recent advancements in deep neural networks have successfully improved a variety of learning problems [1], [2], [3]. For supervised learning, however, massive labeled training data is still the key to learning an accurate deep model. Although abundant labels may be available for a few pre-specified domains, such as ImageNet [4], manual labels often turn out to be difficult or expensive to obtain for every ad-hoc target domain or task. The absence of in-domain labeled data hinders the application of data-fitting models in many real-world problems.

In the absence of labeled data from the target domain, Unsupervised Domain Adaptation (UDA) methods have emerged to mitigate the domain shift in data distributions [5], [6], [7], [8], [9], [10], [11], [12]. It relates to unsupervised learning as it requires manual labels only from the source domain and zero labels from the target. Among the recent work on UDA, a seminal line of work proposed by Long *et al*. [13], [14] aims at minimizing the discrepancy between the source and target domain in the deep neural network, where the domain discrepancy is measured by Maximum Mean Discrepancy (MMD) [13] and Joint MMD (JMMD) [14]. MMD and JMMD have proven effective in many computer vision problems and demonstrated the state-of-the-art results on several UDA benchmarks [13], [14].

Despite the success of previous methods based on MMD and JMMD, most of them measure the domain discrepancy at the *domain* level, neglecting the class from which the samples are drawn. These class-agnostic approaches, hence, do not discriminate whether samples from two domains should be aligned according to their class labels (Fig. 1). This can impair the adaptation performance due to the following reasons. First, samples of different classes may be aligned incorrectly, e.g.both MMD and JMMD can be minimized even when the target-domain samples are misaligned with the source-domain samples of a different class. Second, the learned decision boundary may generalize poorly for the target domain. There exist many sub-optimal solutions near the decision boundary. These solutions may overfit the source data well but are less discriminative for the target.

To address the above issues, we introduce a new *Contrastive Domain Discrepancy (CDD)* objective to enable class-aware UDA. We propose to minimize the intra-class discrepancy, i.e. the domain discrepancy within the same class, and maximize the inter-class margin, i.e. the domain discrepancy between different classes. Considering the toy example in Fig. 1, CDD will draw closer the source and target samples of the same underlying class (e.g. the blue and red triangles), while pushing apart the samples from different classes (e.g. the blue triangle and the red star).

- *Guoliang Kang and Alexander Hauptmann are with the Language Technologies Institute, Carnegie Mellon University, Pittsburgh, PA 15213 USA. E-mail: gkang@andrew, alex@cs.cmu.edu.*
- *Yunchao Wei and Yi Yang are with AAII, University of Technology Sydney, Sydney, NSW 2007, Australia. E-mail: {yunchao.wei, yi.yang}@uts.edu.au.*
- *Lu Jiang is with Google Research, Mountain View, CA 94043 USA. E-mail: lujiang@google.com.*

Manuscript received 24 Dec. 2019; revised 12 Sept. 2020; accepted 4 Oct. 2020. Date of publication 9 Oct. 2020; date of current version 4 Mar. 2022. (Corresponding author: Guoliang Kang.) Recommended for acceptance by R. Feris. Digital Object Identifier no. 10.1109/TPAMI.2020.3029948

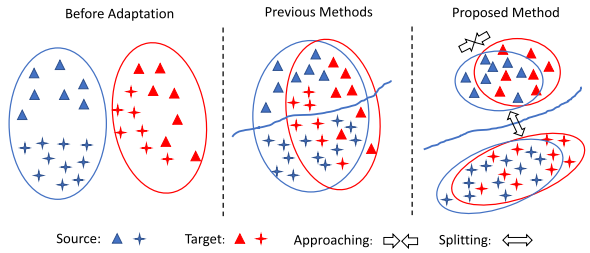

Fig. 1. Comparison between previous domain-discrepancy minimization methods and ours. *Left:* The domain shift exists between the source and target data before adaptation. *Middle:* Class-agnostic adaptation aligns source and target data at the domain-level, neglecting the class label of the sample, and hence may lead to suboptimal solutions. Consequently, the target samples of one label may be misaligned with source samples of a different label. *Right:* Our method performs class-aware alignment across domains. To avoid the misalignment, only the intra-class domain discrepancy is minimized. The inter-class domain discrepancy is maximized to enhance the model's generalization ability.

Unfortunately, to estimate and optimize with CDD, we may not train a deep network out-of-the-box as we need to overcome the following two technical issues. First, we need labels from both domains to compute CDD, however, target labels are unknown in UDA. A straightforward way, of course, is to estimate the target labels by the network outputs during training. However, because the estimation can be noisy, we find it may harm the adaptation performance (see Section 4.3). Second, during the mini-batch training, for a class C, the mini-batch may only contain samples from one domain (source or target), rendering it infeasible to estimate the intra-class domain discrepancy of C. This can result in a less efficient adaptation. The above issues require special design of the network and the training paradigm.

In this paper, we propose *Contrastive Adaptation Network (CAN)* to facilitate the optimization with CDD. During training, in addition to minimizing the cross-entropy loss on labeled source data, CAN alternatively estimates the underlying label hypotheses of target samples through clustering, and adapts the feature representations according to the CDD metric. After clustering, the ambiguous target data (i.e. far from the cluster centers) and ambiguous classes (i.e. containing few target samples around the cluster centers) are zeroed out in estimating the CDD. Empirically, we find that during training, an increasing amount of samples will be taken into account. Such progressive learning can help CAN capture more accurate statistics of data distributions. Moreover, to facilitate the mini-batch training of CAN, we employ the class-aware sampling for both source and target domains, i.e., at each iteration, we sample data from both domains for each class within a randomly sampled class subset. Class-aware sampling can improve the training efficiency and the adaptation performance.

Our method can be readily adopted in the single-source scenario (where only one source domain data exists). For the multi-source scenario (where multiple source domains are available), technically, we can directly apply CAN to train the adaptation model through combining all the source domain data and treating them as a whole. However, in practice, neglecting the discrepancy across source domains may degenerate the adaptation performance. Thus we propose 1) *mutli-source clustering ensemble* and 2) *boundary-sensitive alignment* to exploit the complementary knowledge provided by different source domains, while also avoiding negative domain transfer.

We validate our method on three public UDA benchmarks: Office-31 [9] and VisDA-2017 [15] for the single-source domain adaptation, and DomainNet [16] for the multi-source domain adaptation. The experimental results show that our method performs favorably against the state-of-the-art approaches. Ablation studies are presented to verify the contribution of key components in our framework.

This journal paper extends our previous work [17]. The main extensions are listed as follows: 1) We extend our method to the multi-source adaptation scenario. Further, we propose multi-source clustering ensemble and boundary-sensitive alignment to improve its adaptation performance. 2) We test our method on large-scale multi-source adaptation benchmark DomainNet [16]. The results outperform previous state-of-the-art methods by a large margin, which verifies the effectiveness of our method. 3) We explain more detailed insights and motivations for the system design. 4) We study the failure cases to illustrate the characteristic and the weakness of our method, which inspires further improvement in the future.

In a nutshell, our contributions are as follows,

- We introduce a new discrepancy metric named *Contrastive Domain Discrepancy* (CDD), which can be embedded in the proposed *Contrastive Adaptation Network* (CAN) for performing the class-aware alignment during the end-to-end training.
- We extend our method to the multi-source adaptation scenario. Moreover, we propose using the multi-source clustering ensemble and boundary-sensitive alignment to further improve the adaptation performance of our framework.
- Our method achieves state-of-the-art performance on the single-source adaptation benchmarks (i.e., Office-31 [9], VisDA-2017 [15]) and the challenging large-scale multi-source benchmark (i.e., DomainNet [16]).

2 Related Work

Class-Agnostic Domain Alignment. A common practice for UDA is to minimize the discrepancy between domains to obtain domain-invariant features [13], [14], [18], [19], [20], [21], [22]. For example, Tzeng *et al.* [23] proposed a kind of domain confusion loss to encourage the network to learn semantically meaningful and domain invariant representations. Long *et al.* proposed DAN [13] and JAN [14] to minimize the MMD and Joint MMD distance across domains respectively, over the domain-specific layers. Ganin *et al.* [18] enabled the network to learn domain invariant representations in an adversarial way by back-propagating the reverse gradients of the domain classifier. Unlike these domain-discrepancy minimization methods, our method performs *class-aware* domain alignment.

Discriminative Domain-Invariant Feature Learning. Some previous works pay efforts to learn more discriminative features while performing domain alignment [24], [25], [26], [27], [28], [29]. Adversarial Dropout Regularization (ADR) [26] and Maximum Classifier Discrepancy (MCD) [27] were proposed to train a deep neural network in an adversarial

way to avoid generating non-discriminative features lying in the region near the decision boundary. Similar to us, Long *et al.* [30] and Pei *et al.* [28] took the class information into account while measuring the domain discrepancy. The CDAN [31] conditioned the adversarial training on discriminative information conveyed in the classifier predictions. By enforcing the cluster assumption, the DTA [32] utilized adversarial dropout to learn discriminative features. The GSDA [33] took the class-wise alignment, group-wise alignment and global alignment into consideration during the feature learning. However, our method differs from these methods mainly in two aspects. First, we explicitly model two types of domain discrepancy, i.e. the intra-class domain discrepancy, and the inter-class domain discrepancy. The inter-class domain discrepancy, which has been ignored by most previous methods, is proved to be beneficial for enhancing the model adaptation performance. Second, in the context of deep neural networks, we treat the training process as an alternative optimization over target label hypotheses and features.

Intra-Class Compactness and Inter-Class Separability Modeling. This paper is also related to the work that explicitly models the intra-class compactness and the inter-class separability, e.g.the contrastive loss [34], and the triplet loss [35]. These methods have been used in various applications, e.g. face recognition [36], person re-identification [37], etc. Different from these methods designed for a single domain, our work focuses on adaptation across domains.

Multi-Source Domain Adaptation. Multi-Source domain adaptation has long been investigated by researchers [6], [38], [39]. David *et al.* [6] proposed a uniform convergence learning bound and extended it to the multi-source setting. Based on the assumption that the target distribution can be represented as a mixture of source distributions, Mansour *et al.* [38] proposed the distribution weight combining rule with the theoretical guarantees. Gong *et al.* [40] put up an approach to automatically discover the latent source domains. In this paper, we assume the domain from which each source sample comes is known.

Recently, researchers have resorted to deep models to boost the multi-source adaptation performance [16], [41], [42]. Deep Cocktail Network (DCTN) [41] adopted multi-way adversarial learning to minimize the discrepancy between the target and each of the source domains and used self-training to further improve the performance. Besides minimizing the discrepancy between target and each of the source domains, M^{3}SDA [16] also minimized the discrepancy across different sources during training.

No matter how the source domains are treated (i.e., as a whole or separately), previous methods are mostly class-agnostic, which faces the same issue as the single-source domain adaptation. Thus our proposed class-aware alignment can also benefit the multi-source domain adaptation task. Moreover, we propose using a multi-source clustering ensemble and boundary-sensitive alignment to excavate the potential of CAN to further improve the performance.

3 Methodology

Unsupervised Domain Adaptation (UDA) aims at improving the model's generalization performance on target domain by mitigating the domain shift in data distribution of the source and target domain. UDA can be categorized into single-source domain adaptation and multi-source domain adaptation according to the number of source domains available. To simplify our descriptions, we use single-source domain adaptation as an example to demonstrate the general framework of our method. For the multi-source domain adaptation, we will discuss further in Section 3.5. Formally, given a set of source domain samples $\mathcal{S} = \{(\boldsymbol{x}_1^s, y_1^s), \ldots, (\boldsymbol{x}_{N_s}^s, y_{N_s}^s)\}$, and target domain samples $\mathcal{T} = \{\boldsymbol{x}_1^t, \ldots, \boldsymbol{x}_{N_t}^t\}$, $\boldsymbol{x}^s$, $\boldsymbol{x}^t$ represent the input data, and $y^s \in \{1, 2, \ldots, M\}$ denote the source data label of M classes. The target data label $y^t \in \{1, 2, \ldots, M\}$ is unknown. Thus, in UDA, we are interested in training a network using labeled source data $\mathcal{S}$ and unlabeled target data $\mathcal{T}$ to make accurate predictions $\{\hat{y}^t\}$ on $\mathcal{T}$.

We discuss our method in the context of deep neural networks. In deep neural networks, a sample owns hierarchical features/representations denoted by the activations of each layer $l \in \mathcal{L}$. In the following, we use $\phi_l(\boldsymbol{x})$ to denote the outputs of layer l in a deep neural network Φ_θ for the input $\boldsymbol{x}$, where $\phi(\cdot)$ denotes the mapping defined by the deep neural network from the input to a specific layer.

In the rest of this section, we start our discussions by briefly reviewing the relevant concepts in MMD in Section 3.1. Section 3.2 introduces a new domain discrepancy metric. Finally, Sections 3.3 and 3.4 discuss the objective and training procedure of the proposed contrastive adaptation network (CAN). In Section 3.5, we extend the general CAN framework to the multi-source scenario.

3.1 Maximum Mean Discrepancy Revisiting

In Maximum Mean Discrepancy (MMD), $\{\boldsymbol{x}_i^s\}$ and $\{\boldsymbol{x}_i^t\}$ are *i.i.d.* sampled from the marginal distributions $P(\boldsymbol{X}^s)$ and $Q(\boldsymbol{X}^t)$ respectively. Based on the observed samples, MMD [43] performs a kernel two-sample test to determine whether to accept the null hypothesis $P = Q$ or not. MMD is motivated by the fact that if two distributions are identical, all of their statistics should be the same. Formally, MMD defines the difference between two distributions with their mean embeddings in the reproducing kernel Hilbert space (RKHS), i.e.

$$\mathcal{D}_{\mathcal{H}}(P, Q) \triangleq \sup_{f \sim \mathcal{H}} \left(\mathbb{E}_{\boldsymbol{X}^s}[f(\boldsymbol{X}^s)] - \mathbb{E}_{\boldsymbol{X}^t}[f(\boldsymbol{X}^t)]\right)_{\mathcal{H}}, \tag{1}$$

where $\mathcal{H}$ is a class of functions.

In practice, for a layer l, the squared value of MMD is estimated with the empirical kernel mean embeddings

$$\begin{aligned}\hat{\mathcal{D}}_l^{mmd} = &\frac{1}{n_s^2}\sum_{i=1}^{n_s}\sum_{j=1}^{n_s} k_l(\boldsymbol{\phi}_l(\boldsymbol{x}_i^s), \boldsymbol{\phi}_l(\boldsymbol{x}_j^s)) \\ &+ \frac{1}{n_t^2}\sum_{i=1}^{n_t}\sum_{j=1}^{n_t} k_l(\boldsymbol{\phi}_l(\boldsymbol{x}_i^t), \boldsymbol{\phi}_l(\boldsymbol{x}_j^t)) \\ &- \frac{2}{n_s n_t}\sum_{i=1}^{n_s}\sum_{j=1}^{n_t} k_l(\boldsymbol{\phi}_l(\boldsymbol{x}_i^s), \boldsymbol{\phi}_l(\boldsymbol{x}_j^t)),\end{aligned} \tag{2}$$

where $x^s \in \mathcal{S}' \subset \mathcal{S}$, $x^t \in \mathcal{T}' \subset \mathcal{T}$, $n_s = |\mathcal{S}'|$, $n_t = |\mathcal{T}'|$. The $\mathcal{S}'$ and $\mathcal{T}'$ represent the mini-batch source and target data

sampled from $\mathcal{S}$ and $\mathcal{T}$ respectively. And k_l denotes the kernel selected for the lth layer of deep neural network.

3.2 Contrastive Domain Discrepancy

We propose to explicitly take the class information into account and measure the *intra-class* and *inter-class* discrepancy across domains. The intra-class domain discrepancy is minimized to compact the feature representations of samples within a class, whereas the inter-class domain discrepancy is maximized to push the representations of each other further away from the decision boundary. The intra-class and inter-class discrepancies are jointly optimized to improve the adaptation performance.

The proposed **C**ontrastive **D**omain **D**iscrepancy (CDD) is established on the difference between *conditional* data distributions across domains. Without any constraint on the type (e.g.marginal or conditional) of data distributions, MMD is convenient to measure such difference between $P(\boldsymbol{\phi}(\boldsymbol{X}^s)|Y^s)$ and $Q(\boldsymbol{\phi}(\boldsymbol{X}^t)|Y^t)$, i.e. $\mathcal{D}_{\mathcal{H}}(P,Q) \triangleq \sup_{f\sim\mathcal{H}}(\mathbb{E}_{\boldsymbol{X}^s}[f(\boldsymbol{\phi}(\boldsymbol{X}^s)|Y^s)] - \mathbb{E}_{\boldsymbol{X}^t}[f(\boldsymbol{\phi}(\boldsymbol{X}^t)|Y^t)])_{\mathcal{H}}$.

Supposing $\mu_{cc'}(y,y') = \begin{cases} 1 & \text{if } y=c, y'=c'; \\ 0 & \text{otherwise.} \end{cases}$, for two classes c_1, c_2 (which can be same or different), the kernel mean embedding estimation for squared $\mathcal{D}_{\mathcal{H}}(P,Q)$ is

$$\hat{\mathcal{D}}^{c_1c_2}(\hat{y}_1^t, \hat{y}_2^t, \ldots, \hat{y}_{n_t}^t, \boldsymbol{\phi}) = e_1 + e_2 - 2e_3 \tag{3}$$

where

$$\begin{aligned} e_1 &= \sum_{i=1}^{n_s}\sum_{j=1}^{n_s} \frac{\mu_{c_1c_1}(y_i^s, y_j^s) k(\boldsymbol{\phi}(\boldsymbol{x}_i^s), \boldsymbol{\phi}(\boldsymbol{x}_j^s))}{\sum_{i=1}^{n_s}\sum_{j=1}^{n_s}\mu_{c_1c_1}(y_i^s, y_j^s)} \\ e_2 &= \sum_{i=1}^{n_t}\sum_{j=1}^{n_t} \frac{\mu_{c_2c_2}(\hat{y}_i^t, \hat{y}_j^t) k(\boldsymbol{\phi}(\boldsymbol{x}_i^t), \boldsymbol{\phi}(\boldsymbol{x}_j^t))}{\sum_{i=1}^{n_t}\sum_{j=1}^{n_t}\mu_{c_2c_2}(\hat{y}_i^t, \hat{y}_j^t)} \\ e_3 &= \sum_{i=1}^{n_s}\sum_{j=1}^{n_t} \frac{\mu_{c_1c_2}(y_i^s, \hat{y}_j^t) k(\boldsymbol{\phi}(\boldsymbol{x}_i^s), \boldsymbol{\phi}(\boldsymbol{x}_j^t))}{\sum_{i=1}^{n_s}\sum_{j=1}^{n_t}\mu_{c_1c_2}(y_i^s, \hat{y}_j^t)}. \end{aligned} \tag{4}$$

Note that Eq. (3) defines two kinds of class-aware domain discrepancy, 1) when $c_1 = c_2 = c$, it measures *intra-class* domain discrepancy; 2) when $c_1 \neq c_2$, it becomes the *inter-class* domain discrepancy. To compute the mask $\mu_{c_2c_2}(\hat{y}_i^t, \hat{y}_j^t)$ and $\mu_{c_1c_2}(y_i^s, \hat{y}_j^t)$, we need to estimate target labels $\{\hat{y}_i^t\}$, which will be discussed in Section 3.4.

Based on the above definitions, the CDD is calculated as (The $\hat{y}_1^t, \hat{y}_2^t, \ldots, \hat{y}_{n_t}^t$ is abbreviated as $\hat{y}_{1:n_t}^t$)

$$\begin{aligned} \hat{\mathcal{D}}^{cdd} = & \underbrace{\frac{1}{M}\sum_{c=1}^{M} \hat{\mathcal{D}}^{cc}(\hat{y}_{1:n_t}^t, \boldsymbol{\phi})}_{intra} \\ & - \underbrace{\frac{1}{M(M-1)}\sum_{c=1}^{M}\sum_{c'=1, c'\neq c}^{M} \hat{\mathcal{D}}^{cc'}(\hat{y}_{1:n_t}^t, \boldsymbol{\phi})}_{inter}, \end{aligned} \tag{5}$$

where the intra- and inter-class domain discrepancies will be optimized in the opposite direction.

Note although the estimation of the labels $\{\hat{y}_i^t\}$ can be noisy, the CDD (which is established on MMD) in itself is robust to the noise to an extent. Because MMD is determined by the mean embeddings of distributions in the RKHS, such sufficient statistics are less likely to be severely affected by the label noise, especially when the amount of data is large. We will discuss and verify this in Section 4.3.

3.3 Contrastive Adaptation Network

Deep convolutional neural networks (CNNs) learn more transferable features than shallow methods. But the discrepancy still exists for domain-specific layers. Specifically, the convolutional layers extracting general features are more transferable, while the fully-connected (FC) layers exhibiting domain-specific features should be adapted [13], [14].

In this paper, we start from ImageNet [4] pretrained networks, e.g. ResNet [44], [45], and replace the last FC layer with task-specific ones. We follow the general practice that minimizes the domain discrepancy of last FC layers and fine-tunes the convolutional layers through back-propagation. Then our proposed CDD can be readily incorporated into the objective as an adaptation module over the activations of FC layers. We name our network **C**ontrastive **A**daptation **N**etwork (CAN).

The Overall Objective. In a deep CNN, we need to minimize CDD over multiple FC layers, i.e. minimizing

$$\hat{\mathcal{D}}_{\mathcal{L}}^{cdd} = \sum_{l=1}^{L} \hat{\mathcal{D}}_l^{cdd}. \tag{6}$$

Besides, we train the network with labeled source data through minimizing the cross-entropy loss,

$$\ell^{ce} = -\frac{1}{n'_s}\sum_{i'=1}^{n'_s} \log P_\theta(y_{i'}^s | \boldsymbol{x}_{i'}^s) \tag{7}$$

where $y^s \in \{1, 2, \ldots, M\}$ is the ground-truth of sample $\boldsymbol{x}^s$. $P_\theta(y|\boldsymbol{x})$ denotes the predicted probability of label y with the network parameterized by θ, given input $\boldsymbol{x}$.

Therefore, the overall objective can be formulated as

$$\min_\theta \ell = \ell^{ce} + \beta \hat{\mathcal{D}}_{\mathcal{L}}^{cdd}, \tag{8}$$

where β is the weight of the discrepancy penalty term. Through minimizing $\hat{\mathcal{D}}_{\mathcal{L}}^{cdd}$, the intra-class domain discrepancy is minimized and the inter-class domain discrepancy is maximized to perform class-aware domain alignment.

Note that we independently sample the labeled source data to minimize the cross-entropy loss ℓ^{ce} and those to estimate the CDD $\hat{\mathcal{D}}_{\mathcal{L}}^{cdd}$. In this way, we are able to design a more efficient sampling strategy (see Section 3.4) to facilitate the mini-batch stochastic optimization with CDD, while not disturbing the conventional optimization with cross-entropy loss on labeled source data.

3.4 Optimizing CAN

The framework of CAN is illustrated in Fig. 2. In this section, we focus on discussing how to minimize CDD loss in CAN.

Alternative Optimization (AO). As shown in Eq. (5), we need to jointly optimize the target label hypotheses $\hat{y}_{1:n_t}^t$ and the feature representations $\boldsymbol{\phi}_{1:L}$. We adopt alternative loops to perform such optimization. In detail, at each loop, given current feature representations, i.e. fixing θ, we update target labels through clustering. Then, based on the updated

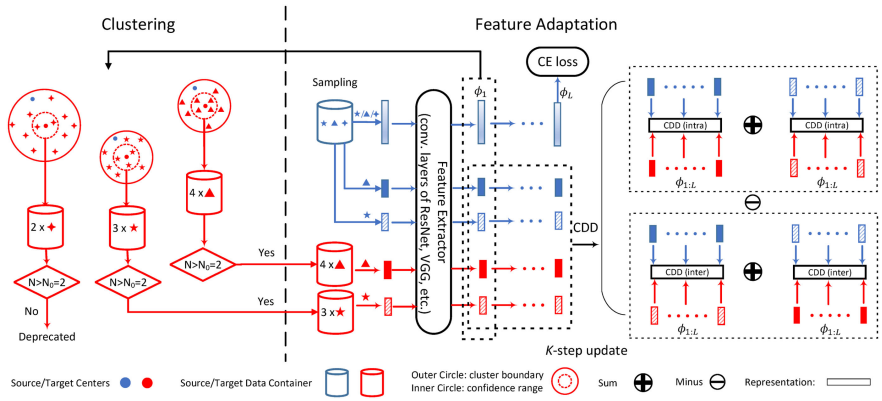

Fig. 2. The training process of CAN. To minimize CDD, we perform alternative optimization between updating the target label hypotheses through clustering and adapting feature representations through back-propagation. For the clustering, we apply spherical K-means clustering of target samples based on their current feature representations. The number of clusters equals to that of underlying classes and the initial center of each class cluster is set to the center of source data within the same class. Then ambiguous data (i.e. far from the affiliated cluster centers) and ambiguous classes (i.e. containing few target samples around affiliated cluster centers) are discarded. For the feature adaptation, the labeled target samples provided by the clustering stage, together with the labeled source samples, pass through the network to achieve their multi-layer feature representations. The features of domain-specific FC layers are adopted to estimate CDD [equation (5)]. Besides, we apply cross-entropy loss on independently sampled source data. Back-propagating with minimizing CDD and cross-entropy loss [equation (15)] adapts the features and provides class-aware alignment. Detailed descriptions can be found in Section 3.4.

target labels $\hat{y}^t$, we estimate and minimize CDD to adapt the features, i.e. update θ through back-propagation.

The reason why we choose clustering to update the target label estimations is that we assume the data from different categories is less likely to concentrate. In other words, at a specific training stage, the peaks of target feature distribution are good representatives for the underlying categories. And clustering can be adopted to discover such peaks. Concretely, we employ the input activations $\phi_1(\cdot)$ of the first task-specific layer to represent a sample. For example, in ResNet, each sample can be represented as the outputs of the global average pooling layer which are also the inputs of the following task-specific layer. Then the spherical K-means is adopted to perform the clustering of target samples and attach corresponding labels.

To perform the K-means clustering, we need to determine 1) the number of clusters and 2) the initial cluster centers. For the first issue, the number of clusters is set to the number of underlying classes M. For the second one, we use source cluster center O^{sc} (which can be directly calculated using source ground-truth labels) to initialize the target cluster center O^{tc} , i.e.

$$O^{tc} \leftarrow O^{sc} = \sum_{i=1}^{N_s} \mathbf{1}_{y_i^s=c} \frac{\phi_1(\boldsymbol{x}_i^s)}{\|\phi_1(\boldsymbol{x}_i^s)\|}, \tag{9}$$

where

$$\mathbf{1}_{y_i^s=c} = \begin{cases} 1 & \text{if } y_i^s = c \\ 0 & \text{otherwise} \end{cases} \tag{10}$$

For the metric measuring the distance between points $\boldsymbol{a}$ and $\boldsymbol{b}$ in the feature space, we apply the cosine dissimilarity, i.e.

$$dist(\boldsymbol{a}, \boldsymbol{b}) = \frac{1}{2}\left(1 - \frac{\langle \boldsymbol{a}, \boldsymbol{b} \rangle}{\|\boldsymbol{a}\|\|\boldsymbol{b}\|}\right). \tag{11}$$

Then the clustering process is iteratively 1) attaching labels for each target samples: $\hat{y}_i^t \leftarrow \arg\min_c dist(\phi_1(\boldsymbol{x}_i^t), O^{tc})$, and 2) updating the cluster centers: $O^{tc} \leftarrow \sum_{i=1}^{N_t} \mathbf{1}_{\hat{y}_i^t=c} \frac{\phi_1(\boldsymbol{x}_i^t)}{\|\phi_1(\boldsymbol{x}_i^t)\|}$, till convergence or reaching the maximum clustering steps.

Note that both the source center initialization and the subsequent K-means clustering contribute to more accurate target label hypotheses. Source center initialization models the shared knowledge across domains, while the K-means clustering on target data takes the target-specific knowledge into consideration.

After clustering, each target sample $\boldsymbol{x}_i^t$ is assigned a label $\hat{y}_i^t$ same as its affiliated clusters. Moreover, ambiguous data, which is far from its affiliated cluster center, is discarded, i.e. we select a subset $\tilde{T} = \{(\boldsymbol{x}^t, \hat{y}^t) | dist(\phi_1(\boldsymbol{x}^t), O^{t(\hat{y}^t)}) < D_0, \boldsymbol{x}^t \in \mathcal{T}\}$, where $D_0 \in [0, 1]$ is a constant.

Moreover, to give a more accurate estimation of the distribution statistics, we assume that the minimum number of samples in $\tilde{T}$ assigned to each class, should be guaranteed. The class which doesn't satisfy such condition will not be considered in current loop, i.e. at loop T_e, the selected subset of classes $\mathcal{C}_{T_e} = \{c | \sum_i^{|\tilde{T}|} \mathbf{1}_{\hat{y}_i^t=c} > N_0, c \in \{1, 2, \ldots, M\}\}$, where N_0 is a constant.

At the start of training, due to the domain shift, it is more likely to exclude partial classes. However, as training proceeds, more and more classes are included. The reason is twofold: 1) as training proceeds, the model becomes more accurate and 2) benefiting from the CDD penalty, the intra-class domain discrepancy becomes smaller, and the inter-class domain discrepancy becomes larger, so that the hard (i.e. ambiguous) classes are able to be taken into account.

Empirically, we find that the above filtering operations bring noticeable improvement when the amount of target data is not sufficiently large. And for the scenario where a large amount of target data is available, without such filtering, we can still achieve comparable adaptation results, because the mean embedding of distribution used to compute the CDD can be more accurately estimated in this case.

Class-Aware Sampling (CAS). In the conventional training of deep neural networks, a mini-batch of data is usually sampled at each iteration without being differentiated by their

classes. However, it will be less efficient for computing the CDD. For example, for class C, there may only exist samples from one domain (source or target) in the mini-batch, thus the intra-class discrepancy could not be estimated.

We propose to use a class-aware sampling strategy to enable the efficient update of the network with CDD. It is easy to implement. We randomly select a subset of classes $\mathcal{C}'_{T_e}$ from $\mathcal{C}_{T_e}$, and then sample source data and target data for each class in $\mathcal{C}'_{T_e}$. Consequently, in each mini-batch of data during training, we are able to estimate the intra-class discrepancy for each selected class.

Algorithm. Algorithm 1 shows one loop of the AO procedure, i.e., alternating between a clustering phase (Step 1-4), and a K-step network update phase (Step 5-11). The loop of AO is repeated multiple times in our experiments. Because the feature adapting process is relatively slower, we asynchronously update the target labels and the network parameters to make the training more stable and efficient.

Algorithm 1. Optimization of CAN at Loop T_e.

Input:
source data: $\mathcal{S} = \{(\boldsymbol{x}_1^s, y_1^s), \ldots, (\boldsymbol{x}_{N_s}^s, y_{N_s}^s)\}$,
target data: $\mathcal{T} = \{\boldsymbol{x}_1^t, \ldots, \boldsymbol{x}_{N_t}^t\}$
Procedure:
1 Forward $\mathcal{S}$ and compute the M cluster centers O^{sc} ;
2 Initialize O^{tc}: $O^{tc} \leftarrow O^{sc}$;
3 Cluster target samples $\mathcal{T}$ using spherical K-means;
4 Filter the ambiguous target samples and classes;
5 **for** $(k \leftarrow 1; k \leq K; k \leftarrow k+1)$ **do**
6 Class-aware sampling based on $\mathcal{C}'_{T_e}$, $\tilde{\mathcal{T}}$, and $\mathcal{S}$;
7 Compute $\hat{\mathcal{D}}_{\mathcal{L}}^{cdd}$ using Eq. (6);
8 Sample from $\mathcal{S}$ and compute ℓ^{ce} using Eq. (7);
9 Back-propagate with the objective ℓ (Eq.(15));
10 Update network parameters θ.
10 **end**

3.5 Multi-Source Contrastive Adaptation Network

For multi-source adaptation task, we have multiple distinct source domains $\{\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_E\}$, where E is the number of source domains. Technically, we can directly adopt CAN to this task, by combining all the source domain data and treating them as a whole. However, this straightforward way neglects the discrepancy across source domains, and makes it infeasible to explicitly "shape" the distributions of different source domains according to their distinct characteristics, which may lead to sub-optimal results. In this paper, we treat each source domain separately and propose using the multi-source clustering ensemble and the Boundary-Sensitive Alignment (BSA) to exploit the complementary knowledge from different source domains and avoid negative domain transfer.

General Framework. The objective for the multi-source adaptation can be formulated as

$$\min_{\theta} \ell = \sum_{e=1}^{E} \ell_e^{ce} + \beta \hat{\mathcal{D}}_{\mathcal{L},e}^{cdd}, \tag{12}$$

The $\hat{\mathcal{D}}_{\mathcal{L},e}^{cdd}$ denotes the CDD loss between the target and the eth source domain. Note that the discrepancy across source domains would also be mitigated because each source domain is separately aligned with the same target domain, which in turn benefits the adaptation.

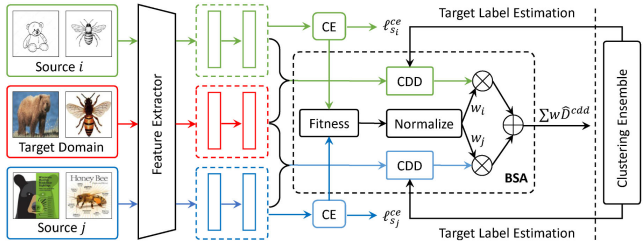

Fig. 3. The framework of multi-source CAN. The feature extractor and the classifier are shared across all the source domains and the target domain. Same as the general CAN framework, the feature update and the clustering are alternatively performed. For the clustering, we first independently cluster target samples based on the centers of each source domain. Then the clustering ensemble is performed to generate more accurate and robust target label estimations. Compared to the general CAN framework shown in Fig. 2, we further introduce the Boundary-Sensitive Alignment (BSA) module to aggregate the CDD losses between target and different sources.

Different from previous methods [16], [41] which split the classifiers for different sources, in our framework, both the feature extractor and the classifier are shared across all the source domains and the target domain. In this way, it is expected that the classifier will be less likely to overfit to one specific domain and thus be more generalizable.

We name the CAN framework for multi-source adaptation tasks as multi-source CAN (MSCAN) which is shown in Fig. 3. Specifically, compared to the general CAN framework (illustrated in Fig. 2), our MSCAN has some unique modules which are listed below.

Boundary-Sensitive Alignment (BSA). At specific stage of training, the extent of data fitting may vary a lot across different source domains. The decision boundary may be biased towards the source domain which is better fitted. Note that such decision boundary is also adopted to classify the target data. Treating each source domain equally during alignment with the target may lead to sub-optimal results. Thus we propose to adopt a *boundary-sensitive* way to assign different weights to the source-target alignments. Specifically, as the cross-entropy loss can indicate how well the model fits the data, the source domain with smaller cross-entropy loss should be emphasized during the domain alignment, i.e., the alignment between target and such source domain should be stronger. Thus the weight for each pair of source-target alignments can be calculated as

$$F_e = 1/\ell_e^{ce} \tag{13}$$

$$w_e = \frac{F_e}{\sum_i F_e}, \tag{14}$$

where F_e denotes the fitness of the eth source domain.

And the objective with weighted CDD loss is

$$\min_{\theta} \ell = \sum_{e=1}^{E} \ell_e^{ce} + \beta w_e \hat{\mathcal{D}}_{\mathcal{L},e}^{cdd}. \tag{15}$$

To make a more accurate and robust estimation for the fitness of source domain data, we adopt moving average

estimation for F_e, i.e.

$$F_e^{t+1} = \frac{1}{(1-\alpha)/F_e^t + \alpha \ell_e^{ce}}, \tag{16}$$

where we set α to 0.9 in our experiments. And t (with a bit of symbol abuse) denotes the number of iterations here.

Through weighted alignment with different source domains, BSA makes the decision boundary better fitted to the target and improves the adaptation performance. Note that different source domains may have different convergence rates. At certain steps, the training of source domains with higher convergence level will be more regularized due to the resulted stronger alignment by BSA. Consequently, in turn, the convergence speed of such source domains will be slowed down. Thus considering the complete training process, the relative fitness of each source domain dynamically changes, which avoids the model being biased towards several specific domains with higher convergence speed in a natural training.

Multi-Source Clustering Ensemble. Recall that for generating the target label hypotheses, we perform sphere k-means clustering at certain steps. The initial centers of target clusters are set to the source centers which can be calculated based on the available source labels. In the multi-source adaptation setting, we can simply treat all the source domain data as a whole to calculate the initial target centers. However, due to the discrepancy across sources, the centers calculated this way may not provide sufficient information to support the subsequent clustering on target. Thus we choose to generate different clustering results initialized by the centers of different source domains. We name each clustering result based on the centers of one specific source domain as one *clustering instantiation*. Established upon these clustering instantiations, ensemble can be performed to utilize the complementary knowledge of various source domains to give more accurate target label estimations.

Considering the clustering ensemble, several alternative solutions can be employed:

1) *Label voting (LV)*. As discussed above, for each target sample, we can obtain a group of label estimations $\{\hat{y}_1^t, \hat{y}_2^t, \ldots, \hat{y}_E^t\}$ based on all the E clustering instantiations. The final target label estimation can be determined by the majority voting.
2) *Average Ensemble (AE)*. In this solution, we first model the probability of each target sample belonging to one specific class c based on the eth clustering instantiation, i.e.,

$$\Pr(\hat{y}_e^t = c) = \frac{\exp\{-\frac{dist(\phi_1(\boldsymbol{x}^t), O_e^{tc}) - \mu_e}{\sigma_e}\}}{\sum_{c'=1}^{M} \exp\{-\frac{dist(\phi_1(\boldsymbol{x}^t), O_e^{tc'}) - \mu_e}{\sigma_e}\}}, \tag{17}$$

where $\mu_e = \frac{1}{M}\sum_{c=1}^{M} dist(\phi_1(\boldsymbol{x}^t), O_e^{tc})$, $\sigma_e = \sqrt{\frac{1}{M-1}\sum_{c=1}^{M}(dist(\phi_1(\boldsymbol{x}^t), O_e^{tc}) - \mu_e)^2}$, O_e^{tc} denotes the target center of class c in the eth clustering instantiation, and the distance measure is the same as that described in Section 3.4. Note that we perform the normalization based on the computed μ_e and σ_e in Eq. (17) to scale the distances to a suitable range and improve their contrast.

Then the target label is set to the class that has the maximum probability averaged over all the ensemble instantiations, i.e.,

$$\hat{y}^t = \arg\max_c \sum_{e=1}^{E} \Pr(\hat{y}_e^t = c). \tag{18}$$

3) *Importance Weighted Ensemble (IWE)*. In the average ensemble, all the clustering instantiations are treated equally. However, in practice, the domain shifts between target domain and source domains are different, i.e., the target could be more closer to part of the source domains than the other. The confidence of the clustering result should decrease as the domain shift increases. In CAN, after clustering, the distance of cluster centers between target and source can reflect some extent of domain shift. And the importance of each clustering instantiation can be evalutated as

$$\tilde{I}^e = \frac{\exp\{\frac{I^e - \mu}{\sigma}\}}{\sum_{e'=1}^{E} \exp\{\frac{I^{e'} - \mu}{\sigma}\}}, \tag{19}$$

where

$$I^e = 1 - \frac{1}{M}\sum_{c=1}^{M} dist(O_e^{tc}, O^{s_e c}), \tag{20}$$

$\mu = \frac{1}{E}\sum_{e=1}^{E} I^e$ and $\sigma = \sqrt{\frac{1}{E-1}\sum_{e=1}^{E}(I^e - \mu)^2}$. And $\frac{1}{M}\sum_{c=1}^{M} dist(O_e^{tc}, O^{s_e c})$ denotes the average distance between the cluster centers of the eth source domain and those of the eth target clustering instantiation, which can reflect the domain shift between source and target to an extent.

Different to the average ensemble, the probabilities of each target sample belonging to a specific class are treated unequally among different clustering instantiations, according to the computed importance $\tilde{I}^e$. And the target label estimation $\hat{y}^t$ is determined by

$$\hat{y}^t = \arg\max_c \sum_{e=1}^{E} \tilde{I}^e \times \Pr(\hat{y}_e^t = c). \tag{21}$$

We employ IWE to ensemble the clustering results in the multi-source adaptation setting. Empirically, we compare the above three solutions in Section 4.3 and find that both AE and IWE outperform the LV method. It is consistent with the experience in ensemble learning, i.e., the voting by soft predictions usually performs better than the voting by hard labels. Moreover, IWE performs the best among three solutions, because it takes the discrepancy between each source domain and the target domain into consideration.

Note that for the multi-source task, the clustering update scheme is the same with the single-source one, i.e., the clustering results are updated every certain steps of training as the feature evolves.

4 Experiments

4.1 Setups

Datasets. We validate our method on three public benchmarks. *Office-31* [9] is a common dataset for real-world domain adaptation tasks. It consists of 4,110 images

TABLE 1
Classification Accuracy (%) for All the Six Tasks of Office-31 Dataset Based on ResNet-50 [44], [45]

Method	A → W	D → W	W → D	A → D	D → A	W → A	Average
Source-finetune	68.4 ± 0.2	96.7 ± 0.1	99.3 ± 0.1	68.9 ± 0.2	62.5 ± 0.3	60.7 ± 0.3	76.1
RevGrad [18], [46]	82.0 ± 0.4	96.9 ± 0.2	99.1 ± 0.1	79.7 ± 0.4	68.2 ± 0.4	67.4 ± 0.5	82.2
DAN [13]	80.5 ± 0.4	97.1 ± 0.2	99.6 ± 0.1	78.6 ± 0.2	63.6 ± 0.3	62.8 ± 0.2	80.4
JAN [14]	85.4 ± 0.3	97.4 ± 0.2	99.8 ± 0.2	84.7 ± 0.3	68.6 ± 0.3	70.0 ± 0.4	84.3
MADA [28]	90.0 ± 0.2	97.4 ± 0.1	99.6 ± 0.1	87.8 ± 0.2	70.3 ± 0.3	66.4 ± 0.3	85.2
CDAN [31]	94.1 ± 0.1	98.6 ± 0.1	**100.0 ± 0.0**	92.9 ± 0.2	71.0 ± 0.3	69.3 ± 0.3	87.7
GSDA [33]	**95.7**	**99.1**	**100.0**	94.8	73.5	74.9	89.7
Ours (intra only)	93.2 ± 0.2	98.4 ± 0.2	99.8 ± 0.2	92.9 ± 0.2	76.5 ± 0.3	76.0 ± 0.3	89.5
Ours (CAN)	94.5 ± 0.3	**99.1 ± 0.2**	99.8 ± 0.2	**95.0 ± 0.3**	**78.0 ± 0.3**	**77.0 ± 0.3**	**90.6**

Our methods named "intra only" and "CAN" are trained with intra-class domain discrepancy and contrastive domain discrepancy, respectively.

belonging to 31 classes. This dataset contains three distinct domains, i.e., images which are collected from the 1) Amazon website (**A**mazon domain), 2) web camera (**W**ebcam domain), and 3) digital SLR camera (**D**SLR domain) under different settings, respectively. The dataset is imbalanced across domains, with 2,817 images in **A** domain, 795 images in **W** domain, and 498 images in **D** domain.

VisDA-2017 [15] is a challenging testbed for UDA with the domain shift from synthetic data to real imagery. In this paper, we validate our method on its classification task. In total there are ∼280k images from 12 categories. The images are split into three sets, i.e. a training set with 152,397 synthetic images, a validation set with 55,388 real-world images, and a test set with 72,372 real-world images.

The above two datasets are adopted to validate the effectiveness of CAN in *single-source domain adaptation* setting.

DomainNet [16] is a large-scale multi-source domain adaptation benchmark. It contains six distinct domains, i.e. Real (photos and real-world images), Infograph (infographic images with specific object), Sketch (sketches of specific objects), Quickdraw (drawings of the worldwide players of game "Quick Draw!"), Clipart (collection of clipart images), and Painting (artistic depictions of objects in the form of paintings). In total, it consists of ∼423.5k images from 345 categories. For each domain, the data is split into a training set and a test set. We test our MSCAN on the DomainNet.

Baselines. We compare our method with previous state-of-the-art methods for single-source and multi-source domain adaptation. 1) *Single-Source*. We compare our method with class-agnostic discrepancy minimization methods: RevGrad [18], [46], DAN [13], and JAN [14]. Moreover, we compare our method with the ones which explicitly or implicitly take the class information or decision boundary into consideration to learn more discriminative features: MADA [28], MCD [27], ADR [26], CDAN [31], DTA [32] and GSDA [33]. The descriptions of these methods can be found in Section 2. For comparison, we re-implement DAN and JAN, and we cite the performance of the other methods reported in their corresponding papers under their optimal parameter setting [18], [26], [27], [28]. 2) *Multi-Source*. We compare our method with previous single-source domain adaptation methods (e.g. RevGrad, DAN, JAN, MCD and SE, *etc*.). We also compare MSCAN with two recent multi-source deep adaptation models, i.e. DCTN [41] and M^{3}SDA [16]. Besides, we compare MSCAN with the general CAN framework which treats all the sources as a whole.

Implementation Details. We use ResNet-50 and ResNet-101 [44], [45] pretrained on ImageNet [4] as our backbone networks. We replace the last FC layer with the task-specific FC layer, and finetune the model with labeled source domain data and unlabeled target domain data. All the network parameters are shared between the source domain(s) and target domain data other than those of the batch normalization layers which are domain-specific. The hyper-parameters are selected following the same protocol as described in [13], i.e. we train a domain classifier and perform selection on a validation set (of labeled source samples and unlabeled target samples) by jointly evaluating the test errors of the source classifier and the domain classifier.

We use mini-batch stochastic gradient descent (SGD) with a momentum parameter of 0.9 to train the network. We follow the same learning rate schedule as described in [13], [14], [18], i.e. the learning rate η_p is adjusted following $\eta_p = \frac{\eta_0}{(1+ap)^b}$, where p linearly increases from 0 to 1. The η_0 is the initial learning rate, i.e. 0.001 for the convolutional layers and 0.01 for the task-specific FC layer. For Office-31 and DomainNet, $a = 10$ and $b = 0.75$, while for VisDA-2017, $a = 10$ and $b = 2.25$. The β selected is 0.3. The thresholds (D_0, N_0) are set to (0.05, 3) for Office-31 tasks **A**→**W** and **A**→**D**. And for the remaining tasks, we don't perform any filtering due to sufficiently large amounts of target data.

4.2 Comparison with the State-of-the-Art

Single-Source. Table 1 shows the classification accuracy on six tasks of Office-31. All domain adaptation methods yield notable improvement over the ResNet model (first row) which is fine-tuned on labeled source data only. CAN outperforms other baseline methods across all tasks, achieving the state-of-the-art performance. On average, it boosts the accuracy of JAN by 6.3 percent and that of GSDA by 0.9 percent.

We visualize the distribution of learned features by t-SNE [48]. Fig. 4 illustrates a representative task **W** → **A**. Compared to JAN, as expected, the target data representations learned by CAN demonstrate higher intra-class compactness and much larger inter-class margin. This suggests that our CDD produces more discriminative target features and substantiates our improvement in Table 1.

Table 2 lists the accuracy of 12 classes on VisDA-2017 with the validation set as the target domain. Our method outperforms the other baseline methods. The mean accuracy of our model (87.2 percent) outperforms the self-ensembling (SE) method [47] (84.3 percent) which wins the first place in the

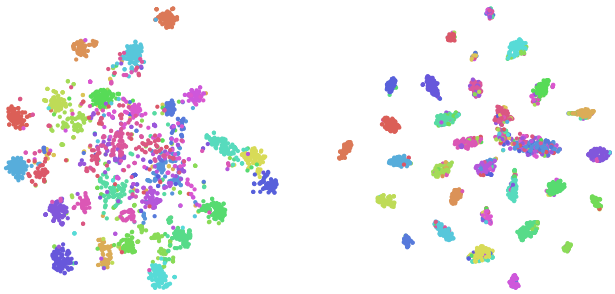

Fig. 4. Visualization with t-SNE for different adaptation methods (best viewed in color). **Left**: t-SNE of JAN. **Right**: CAN. The input activations of the last FC layer are used for the computation of t-SNE. The results are on Office-31 task $\mathbf{W} \rightarrow \mathbf{A}$.

VisDA-2017 competition, by 2.9 percent. It is worth noting that SE mainly deals with UDA by ensemble and data augmentation, which is orthogonal to the topic of this paper and thus can be easily combined to boost the performance further.

Moreover, we also perform adaptation on the VisDA-2017 test set (as the target domain), and submit our predictions to the official evaluation server. Our goal is to evaluate the effectiveness of our proposed technique based on a vanilla backbone (ResNet-101). We choose not to use ensemble or additional data augmentation which is commonly used to boost the performance in the competition. Anyhow, our single model achieves a very competitive accuracy of 87.4 percent, which is comparable to the method which ranks at the second place on the leaderboard (87.7 percent).

From Tables 1 and 2, we have two observations: 1) Taking class information/decision boundary into account is beneficial for the adaptation. It can be seen that MADA, CDAN, MCD, ADR along with our method achieve better performance than class-agnostic methods, e.g. RevGrad, DAN, JAN, *etc*. 2) Our way of exploiting class information is more effective. We achieve better accuracy than CDAN (+2.9 percent), ADR (+12.4 percent), and MCD (+15.3 percent).

Multi-Source. Table 3 shows the classification accuracy on the DomainNet dataset. The "Single-Best" denotes the best accuracy we can achieve among all the possible single-source adaptation results. And the "Multi-Source" means all the source domains are employed to train the model.

It can be seen that our multi-source CAN achieves the best average accuracy compared to all the baselines. Specifically, our method outperforms previous state-of-the-art method M^{3}SDA by a large margin (about 10 percent). Our results are close to the oracle results of ResNet-101 (which can be roughly seen as the upper bound), which also verifies the effectiveness of our proposed method.

TABLE 2
Classification Accuracy (%) on the VisDA-2017 Validation Set Based on ResNet-101 [44], [45]

Method	airplane	bicycle	bus	car	horse	knife	motorcycle	person	plant	skateboard	train	truck	Average
Source-finetune	72.3	6.1	63.4	**91.7**	52.7	7.9	80.1	5.6	90.1	18.5	78.1	25.9	49.4
RevGrad [18], [46]	81.9	77.7	82.8	44.3	81.2	29.5	65.1	28.6	51.9	54.6	82.8	7.8	57.4
DAN [13]	68.1	15.4	76.5	87.0	71.1	48.9	82.3	51.5	88.7	33.2	88.9	42.2	62.8
JAN [14]	75.7	18.7	82.3	86.3	70.2	56.9	80.5	53.8	92.5	32.2	84.5	54.5	65.7
MCD [27]	87.0	60.9	83.7	64.0	88.9	79.6	84.7	76.9	88.6	40.3	83.0	25.8	71.9
ADR [26]	87.8	79.5	83.7	65.3	92.3	61.8	88.9	73.2	87.8	60.0	85.5	32.3	74.8
SE [47]	95.9	**87.4**	85.2	58.6	96.2	95.7	90.6	80.0	94.8	90.8	88.4	47.9	84.3
DTA [32]	93.7	82.2	**85.6**	83.8	93.0	81.0	90.7	82.0	95.1	78.1	86.4	32.1	81.5
Ours (intra only)	96.5	72.1	80.9	70.8	94.6	**98.0**	**91.7**	**84.2**	90.3	89.8	**89.4**	47.9	83.9
Ours (CAN)	**97.0**	87.2	82.5	74.3	**97.8**	96.2	90.8	80.7	**96.6**	**96.3**	87.5	**59.9**	**87.2**

Our methods named "intra only" and "CAN" are trained with intra-class domain discrepancy and contrastive domain discrepancy, respectively.

TABLE 3
Classification Accuracy (%) for the Six Tasks of DomainNet Dataset Based on ResNet-101 [44], [45]

Domain	Method	Target						
		Clipart	Infograph	Painting	Quickdraw	Real	Sketch	Average
Single-Best	Source-finetune	39.6	8.2	33.9	11.8	41.6	23.1	26.4
	RevGrad [18], [46]	37.9	11.4	33.9	13.7	41.5	28.6	27.8
	DAN [13]	39.1	11.4	33.3	16.2	42.1	29.7	28.6
	JAN [14]	35.3	9.1	32.5	14.3	43.1	25.7	26.7
	MCD [27]	42.6	19.6	42.6	3.8	50.5	33.8	32.2
	SE [47]	31.7	12.9	19.9	7.7	33.4	26.3	22.0
	Ours (CAN)	**63.8**	**24.0**	**55.7**	**27.1**	**67.7**	**51.9**	**48.4**
Multi-Source	DCTN [41]	48.6	23.5	48.8	7.2	53.5	47.3	38.2
	M^{3}SDA [16]	58.6	26.0	52.3	6.3	62.7	49.5	42.6
	Ours (CAN)	67.4	25.3	56.2	26.3	72.5	56.2	50.7
	Ours (MSCAN w/o. BSA)	68.5	27.3	57.4	28.1	72.5	58.1	51.9
	Ours (MSCAN w. BSA)	**69.3**	**28.0**	**58.6**	**30.3**	**73.3**	**59.5**	**53.2**
Oracle	ResNet-101	69.3	34.5	66.3	66.8	80.1	60.7	63.0

Our methods named "MSCAN w/o. BSA" and "MSCAN w. BSA" are MSCAN trained without and with boundary-sensitive alignment, respectively.

TABLE 4
The Effect of Alternative Optimization (AO) and CAS

Dataset	w/o. AO	w/o. CAS	CAN
Office-31	88.1	89.1	90.6
VisDA-2017	77.5	81.6	87.2

The mean accuracy over six tasks on Office-31 and the mean accuracy over 12 classes on VisDA-2017 validation set are reported.

Moreover, we have three observations: 1) By comparing the multi-source and the single-best model performance, on the whole, models trained with multiple source domains outperform those trained with data from a single source, which illustrates that training with multiple source domains really helps reduce the negative effect of domain shift. 2) The single-best CAN outperforms all previous single-source methods and is even competitive compared to the models trained with multiple source domains, which further verifies the effectiveness of the general CAN framework. 3) The MSCAN performs noticeably better than the general CAN trained with the combined data from all source domains, demonstrating that neglecting the discrepancy across different source domains may harm the adaptation performance.

4.3 Ablation Studies

Effect of Inter-Class Domain Discrepancy. We compare our method ("CAN") with that trained using intra-class discrepancy only ("intra only"), to verify the merit of the introduced inter-class domain discrepancy measure. The results are shown in the last two rows in Tables 1 and 2. It can be seen that introducing the inter-class domain discrepancy improves the adaptation performance. We believe the reason is that it is impossible to completely eliminate the intra-class domain discrepancy, and maximizing the inter-class domain discrepancy may reduce the possibility of the model overfitting to the source data and benefits the adaptation.

Effect of Alternative Optimization and Class-Aware Sampling. Table 4 examines two key components of CAN, i.e. alternative optimization (or "AO"), and class-aware sampling (or "CAS"). We perform ablation study by leaving-one-component-out of our framework at a time. In Table 4, the method "w/o. AO" directly employs the outputs of the network at each iteration as pseudo target labels to estimate CDD and back-propagates to update the network. It can be regarded as updating the feature representations and pseudo target labels simultaneously. The method "w/o. CAS" uses conventional class-agnostic sampling instead of CAS. The comparisons to these two special cases verify the contributions of AO and CAS in our method.

TABLE 5
Comparison With Different Ways of Utilizing Pseudo Target Labels

Method	A → W	A → D	D → A	W → A	Average
pseudo_0	85.8	86.3	74.9	72.3	79.8
pseudo_1	90.2 ± 1.6	92.5 ± 0.4	75.7 ± 0.2	75.3 ± 0.6	83.4
CAN	94.5 ± 0.3	95.0 ± 0.3	78.0 ± 0.3	77.0 ± 0.3	86.1

The "pseudo_0" means training with pseudo target labels (achieved by our initial clustering) directly. The "pseudo_1" is to alternatively update target labels through clustering and minimize the cross-entropy loss on pseudo labeled target data. In "pseudo_1", the cross-entropy loss on source data is also minimized.

Interestingly, even without alternative optimization, the method "w/o. AO" improves over class-agnostic methods, e.g.DAN, JAN, *etc*. This suggests our proposed CDD in itself is robust to the label noise to some extent, and MMD is a suitable metric to establish CDD (see Section 3.2).

Ways of Using Pseudo Target Labels. The estimates for the target labels can be achieved through clustering, which enables various ways to train a model. In Table 5, we compare our method with two different ways of training with pseudo target labels achieved by the clustering. One way ("pseudo_0") is to fix these pseudo labels to train a model directly. The other ("pseudo_1") is to update the pseudo target labels during training, which is the same as CAN, but to train the model based on the cross-entropy loss over pseudo labeled target data rather than estimating the CDD.

As shown in Table 5, "pseudo_0" leads to a model whose accuracy exactly matches with that of the initial clustering, due to the large capacity of deep neural networks. The "pseudo_1" achieves significantly better results than "pseudo_0", but is still worse than our CAN, which verifies that our way of explicitly modeling the class-aware domain discrepancy makes the model better adapted and less likely to be affected by the label noise.

CDD value During Training. In our training, we generate target label hypotheses to estimate CDD. We expect that the underlying metric computed with the ground-truth target labels would decrease steadily during training until convergence. To do so, during training, we evaluate the *ground-truth* CDD (denoted by CDD-G) for JAN and CAN with the ground-truth target labels. The trend of CDD and the test accuracy during training are plotted in Fig. 5.

As we see, for JAN (the blue curve), the ground-truth CDD rapidly becomes stable at a high level after a short decrease.

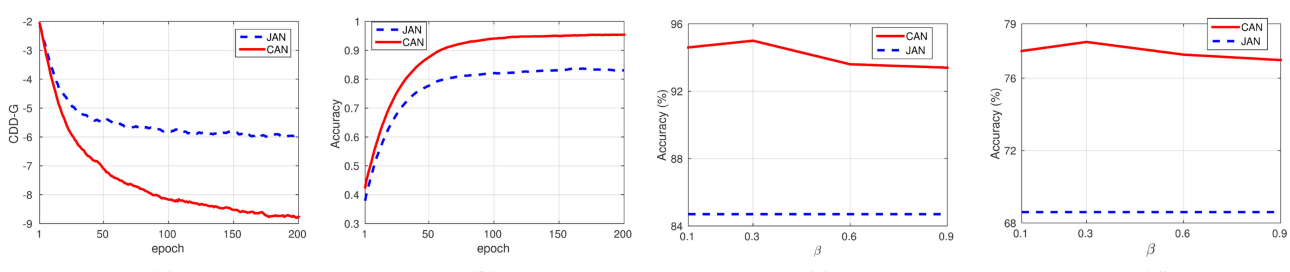

Fig. 5. (a-b) The curve of CDD and accuracy during training on task $\mathbf{A} \rightarrow \mathbf{D}$ of the Office-31 dataset. The "CDD-G" denotes the contrastive domain discrepancy computed with ground-truth target labels. (c-d) The sensitivity of accuracy of CAN to β. The results for $\mathbf{A} \rightarrow \mathbf{D}$ (Left) and $\mathbf{D} \rightarrow \mathbf{A}$ (Right) are illustrated as examples. The trends for other tasks are similar.

TABLE 6
Comparison Among Different Clustering Ensemble Methods Described in Section 3.5

Standards	Method	Average
Single-Best	-	27.1
Multi-Source	LV	26.9
	AE	27.9
	IWE	**28.6**
Source Combine	-	25.9

The numbers showed above are the average clustering accuracy over the six tasks on DomainNet, at the initial training stage. Note that all the ensemble methods start from the same clustering center initialization, but lead to different results of clustering due to the distinct ensemble ways.

This indicates that JAN cannot minimize the contrastive domain discrepancy effectively. For CAN (the red curve), although we can only estimate the CDD using inaccurate target label hypotheses, its CDD value steadily decreases along training. The result illustrates our estimation works as a good proxy of ground-truth contrastive domain discrepancy. And from the accuracy curve illustrated in Fig. 5, we see that minimizing CDD leads to notable accuracy improvement of CAN, compared to JAN.

Hyper-Parameter Sensitivity. We study the sensitivity of CAN to the important balance weight β on two example tasks $\mathbf{A} \rightarrow \mathbf{D}$ and $\mathbf{D} \rightarrow \mathbf{A}$ in Fig. 5. Generally, our model is less sensitive to the change of β. In a vast range, the performance of CAN outperforms the baseline method with a large margin (the blue dashed curve). As the β gets larger, the accuracy steadily increases before decreasing. The bell-shaped curve illustrates the regularization effect of CDD.

Comparison Among Different Multi-Source Clustering Ensemble Methods. We compare different clustering ensemble methods for the multi-source adaptation task in Table 6. It can be seen that among all the ensemble methods, the IWE works the best, which outperforms the single-best clustering result. Both IWE and AE perform better than LV, which is consistent with the common practice in ensemble learning that voting by soft predictions usually leads to better result than voting by hard labels. The "source combine" is inferior to all the ensemble methods, reflecting that the discrepancy across source domains may degenerate the clustering performance.

Effect of Boundary-Sensitive Alignment. In Table 3, we compare the multi-source CANs trained with and without boundary-sensitive alignment (BSA). For all the tasks, the models trained with BSA perform better than those trained without BSA, which illustrates that BSA makes the decision boundary better fitted to the target domain.

Failure Case Study. As illustrated in Fig. 6, we investigate the false predictions on target with the adapted model. Generally, the false predictions can be summarized into two groups: 1) Reasonable failure. In this case, we find that the false predictions are partially due to the concurrency of different objects in the same image. Such predictions cannot be criticized as incorrect. On the contrary, they can reflect the effectiveness of the adapted model from the side. For example, the image showing a man riding a bicycle can be classified as a "person" or a "bicycle". Other false predictions are due to no distinct clues existing in the context or on the object. For these images, it is also confusing for the human to tell the category of the object. 2) Systematic failure. Such failures can be attributed to the model overfitting to the specific shape, texture, pose, or viewpoint. To avoid such false predictions, besides adding more source data with large variety, we can resort to regularizing the training process, e.g.performing abundant data augmentations.

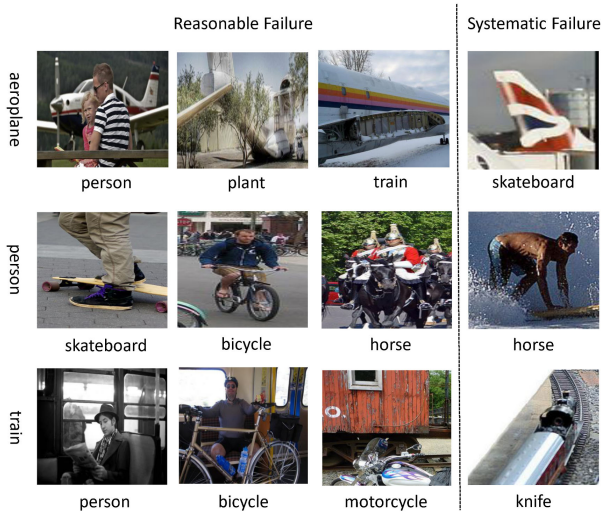

Fig. 6. The illustration of false target predictions. The ground-truth target labels are shown in the leftmost. The label below each image sample is predicted by the CAN. The failure cases can be roughly split into two groups, i.e. the reasonable failure and the systematic failure. The examples shown come from the VisDA-2017 dataset. And similar patterns can be found in other datasets (e.g. Office-31 and DomainNet).

5 Conclusion

In this paper, we proposed the contrastive adaptation network to perform class-aware alignment for UDA. The intra-class and inter-class domain discrepancy are explicitly modeled and optimized through end-to-end mini-batch training. The CAN can be adopted in the single-source and multi-source domain adaptation tasks. For the multi-source setting, we proposed using multi-source clustering ensemble and boundary-sensitive alignment to further improve the adaptation performance. Experiments on real-world benchmarks demonstrate the superiority of our model compared with the strong baselines.

Acknowledgments

This work was supported in part by ARC DECRA DE190101315 and ARC DP200100938.

References

[1] Y. Luo, L. Zheng, T. Guan, J. Yu, and Y. Yang, "Taking a closer look at domain shift: Category-level adversaries for semantics consistent domain adaptation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 2502–2511.

[2] L. Jiang, D. Meng, T. Mitamura, and A. G. Hauptmann, "Easy samples first: Self-paced reranking for zero-example multimedia search," in *Proc. 22nd ACM Int. Conf. Multimedia*, 2014, pp. 547–556.

[3] G. Kang, J. Li, and D. Tao, "Shakeout: A new approach to regularized deep neural network training," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 5, pp. 1245–1258, May 2018.

[4] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2009, pp. 248–255.

[5] S. Ben-David, J. Blitzer, K. Crammer, and F. Pereira, "Analysis of representations for domain adaptation," in *Proc. 19th Int. Conf. Neural Inf. Process. Syst.*, 2007, pp. 137–144.
[6] 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, no. 1–2, pp. 151–175, 2010.
[7] L. Bruzzone and M. Marconcini, "Domain adaptation problems: A dasvm classification technique and a circular validation strategy," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 5, pp. 770–787, May 2010.
[8] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell, "Adversarial discriminative domain adaptation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2017, Art. no. 4.
[9] K. Saenko, B. Kulis, M. Fritz, and T. Darrell, "Adapting visual category models to new domains," in *Proc. Eur. Conf. Comput. Vis. IV*, 2010, pp. 213–226.
[10] J. Hoffman, D. Wang, F. Yu, and T. Darrell, "FCNs in the wild: Pixel-level adversarial and constraint-based adaptation," 2016, *arXiv:1612.02649*.
[11] K. Bousmalis, N. Silberman, D. Dohan, D. Erhan, and D. Krishnan, "Unsupervised pixel-level domain adaptation with generative adversarial networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2017, Art. no. 7.
[12] J. Hoffman *et al.*, "Cycada: Cycle-consistent adversarial domain adaptation," in *Proc. 35th Int. Conf. Mach. Learn.*, 2018, pp. 1989–1998.
[13] M. Long, Y. Cao, J. Wang, and M. Jordan, "Learning transferable features with deep adaptation networks," in *Proc. 35th Int. Conf. Mach. Learn.*, 2015, pp. 97–105.
[14] M. Long, H. Zhu, J. Wang, and M. I. Jordan, "Deep transfer learning with joint adaptation networks," in *Proc. 35th Int. Conf. Mach. Learn.*, 2017, pp. 2208–2217.
[15] X. Peng, B. Usman, N. Kaushik, J. Hoffman, D. Wang, and K. Saenko, "VisDA: The visual domain adaptation challenge," 2017, *arXiv: 1710.06924*.
[16] X. Peng, Q. Bai, X. Xia, Z. Huang, K. Saenko, and B. Wang, "Moment matching for multi-source domain adaptation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 1406–1415.
[17] 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.*, 2019, pp. 4893–4902.
[18] Y. Ganin and V. Lempitsky, "Unsupervised domain adaptation by backpropagation," in *Proc. 35th Int. Conf. Mach. Learn.*, 2015, pp. 1180–1189.
[19] K. Bousmalis, G. Trigeorgis, N. Silberman, D. Krishnan, and D. Erhan, "Domain separation networks," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2016, pp. 343–351.
[20] M. Long, H. Zhu, J. Wang, and M. I. Jordan, "Unsupervised domain adaptation with residual transfer networks," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2016, pp. 136–144.
[21] B. Sun and K. Saenko, "Deep coral: Correlation alignment for deep domain adaptation," in *Proc. Eur. Conf. Comput. Vis.: Part IV*, 2016, pp. 443–450.
[22] 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. Eur. Conf. Comput. Vis.: Part IV*, 2018, pp. 401–416.
[23] E. Tzeng, J. Hoffman, N. Zhang, K. Saenko, and T. Darrell, "Deep domain confusion: Maximizing for domain invariance," 2014, *arXiv:1412.3474*.
[24] O. Sener, H. O. Song, A. Saxena, and S. Savarese, "Learning transferrable representations for unsupervised domain adaptation," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2016, pp. 2110–2118.
[25] P. Haeusser, T. Frerix, A. Mordvintsev, and D. Cremers, "Associative domain adaptation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2017, Art. no. 6.
[26] K. Saito, Y. Ushiku, T. Harada, and K. Saenko, "Adversarial dropout regularization," in *Proc. Int. Conf. Learn. Representations*, 2018.
[27] 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.*, 2018, pp. 3723–3732.
[28] Z. Pei, Z. Cao, M. Long, and J. Wang, "Multi-adversarial domain adaptation," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018, pp. 3934–3941.
[29] J. Wang, W. Feng, Y. Chen, H. Yu, M. Huang, and P. S. Yu, "Visual domain adaptation with manifold embedded distribution alignment," in *Proc. 26th ACM Int. Conf. Multimedia*, 2018, pp. 402–410.
[30] M. Long, J. Wang, G. Ding, J. Sun, and S. Y. Philip, "Transfer feature learning with joint distribution adaptation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2013, pp. 2200–2207.
[31] M. Long, Z. Cao, J. Wang, and M. I. Jordan, "Conditional adversarial domain adaptation," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2018, pp. 1640–1650.
[32] S. Lee, D. Kim, N. Kim, and S.-G. Jeong, "Drop to adapt: Learning discriminative features for unsupervised domain adaptation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 91–100.
[33] L. Hu, M. Kan, S. Shan, and X. Chen, "Unsupervised domain adaptation with hierarchical gradient synchronization," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 4043–4052.
[34] R. Hadsell, S. Chopra, and Y. LeCun, "Dimensionality reduction by learning an invariant mapping," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2006, pp. 1735–1742.
[35] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 815–823.
[36] D. Cheng, Y. Gong, S. Zhou, J. Wang, and N. Zheng, "Person re-identification by multi-channel parts-based CNN with improved triplet loss function," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 1335–1344.
[37] A. Hermans, L. Beyer, and B. Leibe, "In defense of the triplet loss for person re-identification," 2017, *arXiv: 1703.07737*.
[38] Y. Mansour, M. Mohri, and A. Rostamizadeh, "Domain adaptation with multiple sources," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2009, pp. 1041–1048.
[39] L. Duan, D. Xu, and S.-F. Chang, "Exploiting web images for event recognition in consumer videos: A multiple source domain adaptation approach," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2012, pp. 1338–1345.
[40] B. Gong, K. Grauman, and F. Sha, "Reshaping visual datasets for domain adaptation," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2013, pp. 1286–1294.
[41] R. Xu, Z. Chen, W. Zuo, J. Yan, and L. Lin, "Deep cocktail network: Multi-source unsupervised domain adaptation with category shift," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 3964–3973.
[42] H. Zhao, S. Zhang, G. Wu, J. M. Moura, J. P. Costeira, and G. J. Gordon, "Adversarial multiple source domain adaptation," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2018, pp. 8559–8570.
[43] D. Sejdinovic, B. Sriperumbudur, A. Gretton, and K. Fukumizu, "Equivalence of distance-based and RKHS-based statistics in hypothesis testing," *Ann. Statist.*, vol. 41, pp. 2263–2291, 2013.
[44] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.
[45] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," in *Proc. Eur. Conf. Comput. Vis.: Part IV*, 2016, pp. 630–645.
[46] Y. Ganin *et al.*, "Domain-adversarial training of neural networks," *J. Mach. Learn. Res.*, vol. 17, no. 1, pp. 2096–2030, 2016.
[47] G. French, M. Mackiewicz, and M. Fisher, "Self-ensembling for visual domain adaptation," in *Proc. Int. Conf. Learn. Representations*, 2018.
[48] L. V. D. Maaten and G. Hinton, "Visualizing data using t-sne," *J. Mach. Learn. Res.*, vol. 9, pp. 2579–2605, 2008.

Guoliang Kang received the PhD degree from the University of Technology Sydney in 2019. He is currently a postdoctoral research associate at Carnegie Mellon University.

Lu Jiang is currently a senior research scientist at Google Research.

Yunchao Wei is currently an assistant professor at the University of Technology Sydney.

Yi Yang is currently a professor at the University of Technology Sydney.

Alexander Hauptmann is currently a professor at Carnegie Mellon University.